Algorithmic Robot Planning
Homework 1

Andrew Elashkin, Yonatan Sommer

November 2022

Q2 Properties of Minkowski Sums and Euler's theorem

a.

In the lectures we defined the Minkowski sum of two sets $S_1 \subset \mathbb{R}^2$ and $S_2 \subset \mathbb{R}^2$ as:

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

Where $p + q$ is the vector sum of p and q. Also,

$$A \cup B = \{x : x \in A \text{ or } x \in B\}$$

Now, we are asked to prove that given sets A, B and C

$$A \oplus (B \cup C) = (A \oplus B) \cup (A \oplus C)$$

by definition above

$$A \oplus (B \cup C) = \{p + q : p \in A, q \in B \cup C\} = \{(p \text{ or } r) + (q \text{ or } s) : p, r \in A, q \in B, s \in C\}$$

$$= \{p + q : p \in A, q \in B\} \cup \{r + s : r \in A, s \in C\} = (A \oplus B) \cup (A \oplus C)$$

b.

b1.

The Minkowski Sum of two points is the point representing the vector sum of the two vectors represented by the two points.

b2.

The Minkowski Sum of a point and a line is the same line (moved in the plane depending on placement of the shapes and the origin).

b3.

The Minkowski Sum of two line segments is a parallelogram with the line segments as its adjacent edges. If the line segments are parallel, the parallelogram collapses to be a line segment with the same slope that is the sum of the lengths.

b4.

The Minkowski Sum of two disks is a disk with a radius that is the sum of the two radii.

c.

When referring to a planar graph in this section we will be considering finite connected planar graphs as we did in the lecture. We need to prove for planar graph with at least 3 vertices:

$$E \leq 3V - 6 \tag{1}$$

From Euler's formula for planar graphs we know that for any planar graph the following holds:

$$V - E + F = 2 \tag{2}$$

Proof. Let a planar graph G with $V \geq 3$.
If G has only 2 edges since it is a simple graph it must have exactly 3 vertices. Obviously $2 \leq 3 \cdot 3 - 6 = 3$ so (1) holds.
Otherwise G has at least 3 edges.
Define an assignment of the edges to the faces:
Each edge is assigned the faces that touch it, and if an edge touches the same face on both sides (one of its vertices is a leaf on the face boundary), the edge is assigned twice to that face.
Since G is a simple planar graph, each face is bounded by at least 3 edges, and each edge is either bounded by exactly 2 faces or touches a face on both sides and is assigned twice. Therefor each edge is assigned exactly twice and each face is assigned at least 3 edges. That is equivalent to saying there are at least 3 edges per 2 faces or $F \leq \frac{2}{3}E$. Plugging that result into (2) we get $2 \leq V - E + \frac{2}{3}E \Rightarrow \frac{1}{3}E \leq V - 2 \Rightarrow$ (1). ■

Q3 Exact Motion Planning for a Diamond-Shaped Robot

3.2

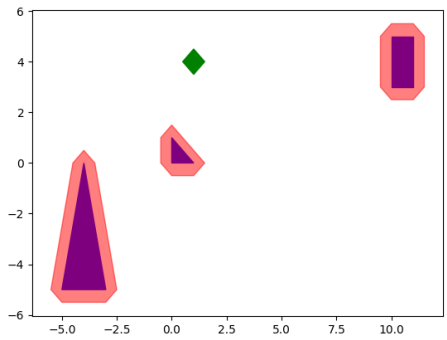

Figure 1: Minkowski Sum of convex obstacles

In our implementation we used convex hull to compute the Minkowski sums, and that implementation has an assumption that all the polygons are convex. In the case that the polygons are

not convex, the algorithm will still treat them as if they were filled to be convex and we will get the wrong Minkowski sum, that will be larger than the correct one. You can see an example of such computation: in Figure 1 the middle obstacle is a convex triangle and you can see it's Minkowski sum drawn around it. In Figure 2 the middle obstacle is non-convex and a subset of points from the triangle from the previous figure, although since our algorithm essentially "fills in" the shape to be convex when making the calculation, the resulting Minkowski sum is identical to the triangle's even though it should be smaller.

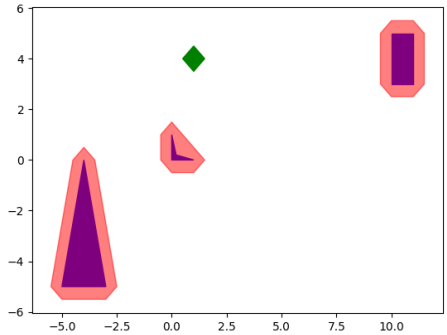

Figure 2: Minkowski Sum of non-convex obstacle

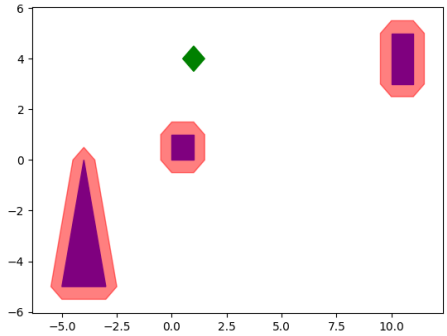

Figure 3: C-space representation of the obstacles

3.3

The visibility graph of the given obstacle set without start and end points can be seen on Figure 4. The computational complexity of our solution is $O(n^3)$. We derive all vertices from the set of obstacles (takes $O(n)$ to go over the set), and then for every two distinct points in the vertex list (complexity $O(n(n-1)/2)$) we calculate if it intersects with any edge of a polygon from the input set. Since we assume that input polygons are simple and convex, their Minkowski sums are also

convex and at most we have $n-2$ edges to check intersection with for each vertex (complexity $O(n)$). So, total complexity is $O(n + \frac{n(n-1)}{2} * n) = O(n^3)$.

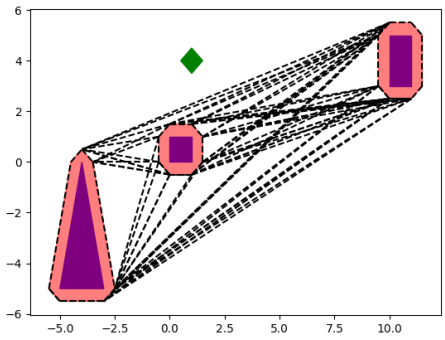

Figure 4: Visibility Graph

3.4

We implemented a lazy version of Dijkstra that allows edges to be added to the priority queue when the same edge still exists in the queue. This can result in at most $|E| \cdot |E| = |E|^2$ edges in the queue for a bottleneck of $O(|E|^2 \log |E|^2) = O(|E|^2 \log |V|)$ complexity. Obviously this can be improved to $O(|E| \log |V|)$ as is the best complexity for Dijkstra. You can see the shortest path constructed in Figure 5.

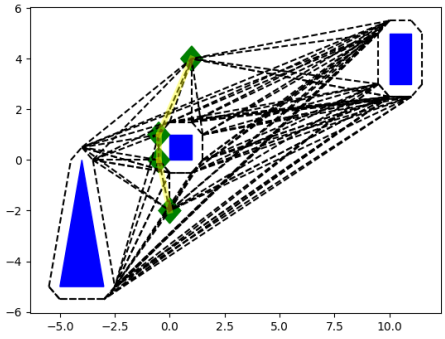

Figure 5: Shortest Path