Chapter 1

Polygons

(with material from [1], [2], and [4], pictures are missing)

Many (real-world) application aim for problems of the following kind. Given two geometric objects:

- Do they intersect?
- Form their intersection?

The former is a geometric predicate, similar to the ones we considered so far, with the difference that the output value is boolean, and not three-valued. The later operation is new. We say it is a geometric *construction*.

Both operations are trivial for points, as they are zero-dimensional. Non-trivial examples already around in low dimensions: lines, line segments, polygons, circles, disks, wedges, polyhedra, spheres, and balls - and many more. We will see that the decision problem is often easier than the constructive version. In addition, the construction has a new geometric object as outcome that might be processed further in the current algorithm or another one.

In Lecture **??** we have seen how to construct the convex hull of a set of points. The convex hull forms a convex polygon, the kind of objects we are often dealing with in this lecture. Polygons are rather fundamental in computational geometry, computer graphics, pattern recognition, robotics.

A polygon is a plane figure. It is bounded by a closed path that is composed of finitely many straight line segments. The segments of the polygon are usually given implicitly, that is, by a cyclic sequence of n points (vertices/corners) $P := (p_0, \ldots, p_{n-1})$. We have $p_n = p_0$. It has *size* n. Segments $S(P) := \{\overline{p_i p_{i+1 \bmod n}}\}$ are the edges of the polygon (or also called sides). The segment $\overline{p_i, p_j}$ are called the diagonals of the polygon. Usually a polygon consists of at least three points and no three consecutive (modulo n) points are collinear.

A polygon can be *convex*, that is, each line intersecting it crosses the polygon's interior, crosses its boundary exactly twice. It is non-convex if

there is a line that crosses its boundary more than twice. It is called *simple* if the edges only intersect at their endpoints. Then the polygon and a closed disk are topologically equivalent. Or interpreting as a graph: The vertices of the graph $G(P, S(P))$ all have degree 2. A simple polygon splits the plane into a bounded and an unbounded region (Jordan's theorem).

We say that a polygon is *weakly simple* if the chain does not cross itself. Note that this is stronger than saying that the edges are interior disjoint. The bounded set of a weakly simple polygon must not be a 2-manifold (that is, a set where the for each point there exists an ϵ-neighborhood that is topologically equivalent to a disc or a half-disk). Or more intuitively, we can have more than one two-dimensional bounded region induces by the sides of the polygon.

A (weakly) simple polygon has also an orientation, resulting from the fact that the chain does not have self-crossings. When we traverse the chain, the bounded region(s) are either all to the left or all to the right of the chain. In the former case we say that P is positively (counter-clockwisely) oriented and the bounded region is the positive area of P. We sometimes also refer to P as the positive part (and not the chain itself).

Polygons imply a set of geometric decisions and constructions. Here are a few examples:

- Is a polygon convex?
- Is a polygon (weakly) simple?
- What is the (signed) area of a polygon?
- Construct a triangulation of a polygon!
- What is the orientation of a polygon?
- Is a point contained in the positive or negative area of a polygon, or on its boundary?
- Construct the complement of a polygon?
- Do two polygons intersect (in a two-dimensional set)?
- Is one polygon contained in another('s interior)?
- Construct the intersection/union/(symmetric)difference of two polygons!
- Construct a triangulation of a polygon!

We next discuss how to decide/compute those problems. Some of them will use as a basic tool a method to decide whether a set of line segments intersect, or to compute their intersections. For example, polygon simplicity test and polygon intersection is linear-time transformable to those basic test.

1.0.1 Intersections

Obviously, there is a difference in computing whether two polygons intersect, and really computing the intersecting set of the two. Both problems arise in robotics, cutting, or computer games. The maximum number of intersections depends on the type of the involved polygons p and q with $|p| = n$ and $|q| = m$. If both are convex the number of intersections is bounded by $\min(2n, 2m)$, if one is convex, the number is bounded by $\max(2n, 2m)$, and otherwise there can be up to mn intersections.

Exercise 1. *Proof the bounds.*

Intersection detection is algorithmically faster than intersection construction, especially when we allow pre-processing that makes use of the structure of polygons, but not their geometric coordinates. For example, with $O(1)$ pre-processing time, it can be decided in $O(n)$ whether two convex (two simple) polygons intersect. Pre-processing time of $O(n)$ ($O(n \log n)$ reduces query time to $O(\log n)$ ($O(m \log^2 n)$), where m is the complexity of the minimum link witness for the intersection or disjointness of the two polygons, where $m \leq n$ always holds. Space requirement is always linear.

1.1 Triangulation

We will not cover triangulation of polygons and refer the reader to [1, Chapter 3] for more details. We only state that a simple polygon has $n - 3$ diagonals and $n - 2$ triangles. And: Every simple polygon admits a diagonal that breaks the polygon into two subpolygons, neither one with more than $\lceil 2n/3 + 1 \rceil$ vertices.

1.2 Signed Area

See [3, Chapter 10.8].

1.3 Orientation

See [3, Chapter 10.8].

1.4 Point Containment

Is an exercise. See [3, Chapter 10.8].

1.5 Complement

See [3, Chapter 10.8]. (Simple, reversal of points).

1.6 (Regularized) Boolean operations

.

Will be discussed after we introduce arrangements in Lecture **??**.

Bibliography

[1] Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark H. Overmars. *Computational geometry*. Springer, Berlin [u.a.], 3., ed. edition, 2008.

[2] Jacob E. Goodman and Joseph O'Rourke. *Handbook of discrete and computational geometry*, volume - of *Discrete mathematics and its applications*. Chapman & Hall/CRC, Boca Raton, 2nd. ed. edition, 2004.

[3] Kurt Mehlhorn and Stefan Näher. *LEDA: A Platform for Combinatorial and Geometric Computing*. Cambridge University Press, Cambridge, UK, 2000. Available online at `http://www.mpi-inf.mpg.de/~mehlhorn/LEDAbook.html`.

[4] Mariette Yvinec. 2d triangulations. In CGAL Editorial Board, editor, *CGAL User and Reference Manual*. 3.5 edition, 2009.