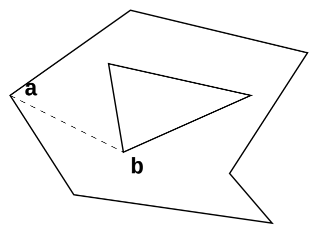
CSCE 620: Computational Geometry
Homework 3 Solutions
Fall 2009

1. *Problem 3.1. Prove that any polygon admits a triangulation, even if it has holes. Can you say anything about the number of triangles in the triangulation?*

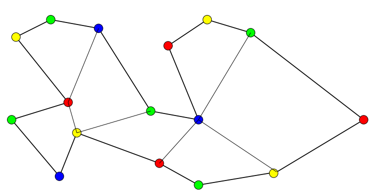
   A polygon with holes can be triangulated by first tranforming it into a simple polygon without holes, and then triangulating it. In particular, a hole can be removed by adding a diagonal from one of the hole vertices to a vertex of the enclosing polygon, e.g., the diagonal from vertex a to vertex b in the figure. This diagonal can be seen as adding two additional vertices to the simple polgonal since both sides of the new diagonal represent external boundaries of the polygon.

   This process is repeated for all h holes in the original polygon. The resulting simple polygon polygon will have $n + 2h$ vertices. Hence, by Theorem 3.1, the triangulated polygon will have $n + 2h - 2$ triangles.

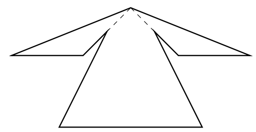
2. *Problem 3.4. Suppose that a simple polygon $\mathcal{P}$ with n verices is given, together with a set of diagonals that partitions $\mathcal{P}$ into convex quadrilaterals. How many cameras are sufficient to guard $\mathcal{P}$? Why doesn't contradict the Art Gallery Theorem?*

   Since the quadrilaterals are convex, each one can be guarded by on guard placed on any of its vertices. Also, the entire polygon can be guarded if each quadrilateral is guarded. Now, we can 4-color of $\mathcal{P}$ as follows (see figure). Select one quadrilateral of $\mathcal{P}$ and 4-color it. Then, iteratively, 4-color one of quadrilaterals that is adjacent to the already colored quadrilaterals. Continue until all vertices of $\mathcal{P}$ are colored. Colored in this manner, only 4 colors will be needed to color $\mathcal{P}$, and each quadrilateral in $\mathcal{P}$ will have one vertex colored with each of the 4 available colors. Hence, $\mathcal{P}$ will be guarded if we select all vertices from any of the 4 color classes. By selecting the smallest color class, we will need $\lfloor n/4 \rfloor$ cameras.

   The Art Gallery Theorem (Theorem 3.2) states that $\lfloor n/3 \rfloor$ cameras are occassionally necessary and always sufficient to cover any simple polygon. The above construction does not violate the Art Gallery Theorem because not all polygons can be decomposed into convex quadrilaterals (whereas all polygons can be triangulated).

3. *Problem 4.1. Give an example of an object that cannot be manufactured with a two-piece mold, but can be manufactured with three-piece mold.*



There is no way to partition the polygon in the figure into two pieces such that both pieces would have a mold with a feasible extraction direction. However, it can be manufactured with three molds, e.g., the partition shown by the dashed lines.

4. *Problem 4.3. Suppose that, in the 3-dimensional casting problem, we do not want the object to slide along a facet of the mold when we remove it. How does this affect the geometric problem (computing a point in the intersection of half-planes) that we derived?*

If the object cannot slide along a facet, then we need to disallow removal directions that would do this. These directions are exactly those in which the inequalities representing the allowable removal directions hold with equality, i.e.,

$$a_1x + b_iy = c_i.$$

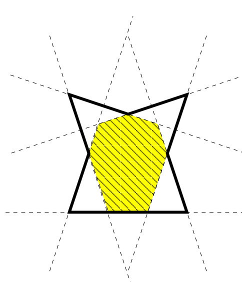
Hence, we only allow removal directions that are strictly less than (<), i.e.,

$$a_1x + b_iy < c_i.$$

This means that the candidate removal directions (the feasible region of the linear program) do not include the bounding lines l_k of the candidate halfplanes h_k, but instead lie strictlly in the interior of the intersection of the candidate halfplanes.

5. *Problem 4.15. A simple polygon $\mathcal{P}$ is called* star-shaped *if it contains a point q such that for any point p in $\mathcal{P}$ the line segment $\overline{pq}$ is contained in $\mathcal{P}$. Give an algorithm whose expected running time is linear to decide whether a simple polygon is star-shaped.*

First note that a polygon is star shaped if and only if the intersection of the halfplanes defined by the polygon's edges is non-empty. See figure.



Hence, we can define a linear programming problem where the feasible region is defined by the intersection of the halfplanes containing the edges, and we can use 2D linear programming algorithm to determine if the feasible region is non-empty in $O(n)$ time, where n is the number of vertices in the polygon.