CS 274
Computational Geometry (Spring 2015)
Homework 2

All the homework rules from Homework 1 remain in effect. Please reread the first page of Homework 1 if you don't remember the policies. All homeworks in this class should be treated like take-home exams.

Homework 2 is **due at the start of class (5:40 pm) on Wednesday, March 4**.

[1] Voronoi overlays (3 points). Let V and W be two point sets in the plane, together having n points. Consider the overlay of their Voronoi diagrams Vor(V) and Vor(W).

(i) Give an example showing how to choose sets V and W so the overlay has $\Theta(n^2)$ faces. (Make sure it is clear how to generalize your example as $n \to \infty$.)

(ii) Prove that, for any point sets V and W, the overlay of their Voronoi diagrams has only $O(n)$ *unbounded* faces.

[2G] Security camera (9 points). Let X be a PSLG with n segments, which represent the walls of a gallery. Some of the walls separate a bounded "inside" region from an unbounded "outside" region where there are no walls. Inside the gallery, there may be walls (non-crossing segments) anywhere, some of which are just there to hang paintings on. Stated more rigorously, there is a simple polygon P, not necessarily convex, such that every edge of P is a segment in X, and for every segment $s \in X$, $s \subset P$.

A fixed, rotating security camera is attached to the ceiling at a point q in the gallery's interior (i.e. $q \in$ interior P). The visibility between two points q and p is *occluded* if there is a segment $s \in X$ such that s intersects the line segment qp and the points p and q lie on opposite sides of s. (If q or p is collinear with s, then s does not occlude the visibility between them. If the camera is placed on a wall, it can see on both sides of the wall.) A point p is *visible* from q if no segment in X occludes the visibility between them.

(i) The subset of P visible from q is a simple polygon Q (not necessarily convex); see the figure on the next page. Give (in English) an efficient algorithm that outputs Q's edges in counterclockwise order, and analyze your algorithm's running time. You do not need to prove your algorithm's correctness (though it does need to be correct). Algorithms slower than $O(n \log n)$ time will only receive partial credit. Hint: there are ways to sweep the plane that don't involve moving a line linearly.

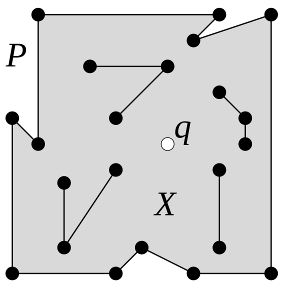
(ii) The gallery is displaying eight masterpieces by the renowned artist Florent de Microscopie, famed for his paintings the size of points. Each of the eight paintings is hung at a different point inside the gallery. Give an efficient algorithm to determine whether there is a single point $q \in P$ from which a fixed rotating security camera can monitor all eight paintings, and to output one such point if at least one exists. Analyze its worst-case running time in term of n. (Don't try to give an output-sensitive bound; the output size is one.) Unduly slow algorithms (or fast algorithms with slow analyses) will only receive partial credit, but you should not expect to achieve $O(n \log n)$ time. (Note: even if you didn't solve part (i), you can still answer this question by assuming you know a solution to part (i).)

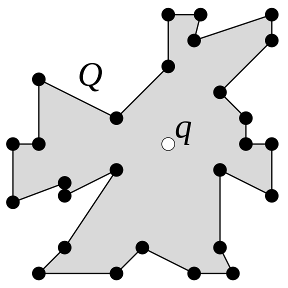
[3] Cocircularities (6 points). Let V be a vertex set in the plane. V may have subsets of four or more cocircular vertices, so V may have many Delaunay triangulations.

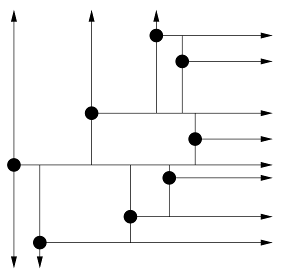
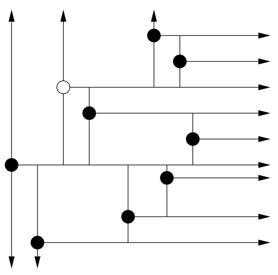
(i) Prove that it is possible to transform a triangulation of a convex polygon to any other triangulation of the same polygon by a sequence of flips.

(ii) Prove that it is possible to flip from any Delaunay triangulation of V to any other Delaunay triangulation of V, such that every intermediate triangulation is also Delaunay.

(iii) Prove that *every* Delaunay triangulation of V maximizes its minimum angle. This means that there is no triangulation of V whose smallest angle is greater.






the subdivision

after point insertion

[4] A randomized incremental algorithm (8 points). Given a set P of n points in the plane, where no two points have the same x- or y-coordinate, define a planar subdivision as follows. Imagine all the points are sorted lexicographically. For each point in this order, shoot one bullet up, one bullet down, and one bullet to the right until they hit existing segments, then add these three bullet-path segments/rays to the subdivision. Let's investigate an algorithm for building this structure in expected $O(n \log n)$ time.

(i) Give tight upper bounds (numerical, not asymptotic) on the number of vertices, edges, and faces of the subdivision as a function of n. Vertices include points in P, segment intersections, and one "vertex at infinity." Faces include unbounded faces. Assume $n > 1$. No explanation is required.

(ii) Describe an efficient algorithm to add a new point to the subdivision and restore the proper subdivision structure. The new point may have an arbitrary x-coordinate (as long as it's different from the other points), but the subdivision must be updated as if the points had been processed in lexicographic order. (See the figure.) Assume the data structure is a DCEL, and you know which face contains the point (we'll handle point location in part (iv)).

(iii) Prove that if a set of points are inserted in random order (starting from scratch), the expected number of structural changes to the subdivision is $O(1)$ per insertion. Give the smallest upper bounds you can (that do not depend on n) on the expected number of new vertices created, the expected number of old vertices eliminated, the expected number of new faces created, and the expected number of old faces eliminated during a vertex insertion.

(iv) Again suppose the points are inserted in random order. Explain how to locate each point in expected $O(\log n)$ time so it may be inserted. (You will need to augment the DCEL with additional data structures.) Prove that your method is that fast.

[5G] Constrained Delaunay triangulations (4 points). Sometimes you want to construct a structure like a Delaunay triangulation, but you need to enforce the presence of certain edges in the triangulation—for instance, the boundaries of an object or domain you're triangulating.

Let X be a PSLG. Define the visibility between two points as in problem [2]. A triangle t is *constrained Delaunay* if its vertices are vertices in X, the interior of t intersects no segment in X, and the circumcircle of t encloses no vertex of X that is visible from any point in the interior of t.

Let t be a triangle that satisfies the first two of those three conditions. Let q be a point in the interior of t. Prove that if no vertex of X inside t's circumcircle is visible from q, then no vertex of X inside t's circumcircle is visible from *any* point in the interior of t (so t is constrained Delaunay).

Hint: Most people forget to consider the possibility that a segment s_1 blocks the visibility (from q) of an endpoint (inside t's circumcircle) of a segment s_2, which blocks the visibility of an endpoint of a segment s_3, which blocks the visibility of an endpoint of s_1; but one of these endpoints is visible from some point in the interior of t. Make sure your proof (directly or indirectly) rules out this possibility.