15-456 Computational Geometry, Spring 2013

Homework 2 (125 pts) **Due: Feb. 8**

Question	Points	Score
1	20	
2	25	
3	20	
4	30	
5	30	
Total:	125	

(20) 1. **Hull Problem** ($20 = 10 + 10$)

Prove the following two claims:

(a) Let $\ell_1, \dots, \ell_k$ be the line segments on the Upper Hull, ordered from leftmost segment to rightmost segment. The slope of ℓ_i is strictly greater than the slope of ℓ_{i+1} for all $i \in \{1, \dots, k-1\}$ (assuming that no two line segments are parallel).

(b) Recall the variant of the Graham scan algorithm discussed in Lecture 4. Let H_i be the value for H after v_i is pushed onto H. Let v_j be the vertex that succeeds v_i in the upper hull of the first i vertices ($v_j = H.second$ after v_i is pushed). Let v_k be a vertex whose x-coordinate lies between the x-coordinates of v_i and v_j (if such a v_k exists). Then, ORIENTATION$(p_i, p_j, p_k) > 0$.

(25) 2. **Convex Hull** ($20 = 10 + 10 + 5$)

Another algorithm that can be used to compute the convex hull is the divide and conquer algorithm. The general flavor of the algorithm is as follows: divide the input set S into k sets of size m $\{S_1, \dots, S_k\}$, compute the convex hull of the smaller sets (using whichever method you would like), then merge the results. We will work on understanding the complexity of the merge step.

(a) Let $p \in \mathbb{R}^2$, and let P be a convex polygon in $\mathbb{R}^2$ with m vertices. A tangent from p to P is a directed line ℓ from p to $q \in P$ such that for all $x \in P$, x lies to the left of ℓ. Show that ℓ is uniquely defined and give an algorithm to find ℓ in $O(\log m)$ time. Be sure to explain how the polygon P is stored.

(b) Let p_0 be the rightmost point of all points in S. Give an algorithm to find the next edge of the convex hull of S in $O(k \log m)$ time, assuming that we have computed the convex hulls of $S_1, \ldots, S_k$.

(c) If h is the total number of vertices on (the boundary of) the convex hull of S, what is the time complexity of computing the convex hull of S given the k convex hulls of the subsets $\{S_i\}$?

(20) 3. **Line Intersection** Write the equation for the lines through the following points in the Euclidean plane:

(a) ℓ_1 through $p_1 = (a, b)$ and $p_2 = (c, d)$.

(b) ℓ_2 through $p_3 = (q, r)$ and $p_4 = (s, r)$.

Assuming that the lines are directed p_1 to p_2 and p_3 to p_4 respectively, what are the homogeneous coordinates for the lines ℓ_1 and ℓ_2?

Compute the homogeneous coordinates for the intersection between ℓ_1 and ℓ_2. (HINT: what is the intersection if ℓ_1 and ℓ_2 are parallel in the Euclidean plane?)

(30) 4. **Topological Sweep** (5 points each). The incremental construction to compute an arrangement takes $\Omega(n^2)$ time and $\Omega(n^2)$ space. However, if we are not interested in storing the whole arrangement, we can visit each vertex in an arrangement using just $O(n)$ space. Suppose we have a set of non-vertical lines in general position (no two parallel, no three through a common point) and a *topological line* that crosses each line exactly once. A topological line is the image of a continuous bijection $p\colon [-\infty, \infty] \to \mathbb{R}^2$ that separates the plane into two regions; see Figure 1. Given two points $p_1, p_2 \in im(p)$, we can find the preimages $t_1 = p^{-1}(p_1)$ and $t_2 = p^{-1}(p_2)$. If $t_1 < t_2$, we say that p_1 lies *above* p_2 and that p_2 lies *below* p_1. The following steps will visit all intersections in this arrangement that are to the right of that path.

(a) Let $p_1, p_2, \ldots, p_n$ be the n intersection points between p and the lines of the arrangement. Without loss of generality, we will assume that the points are ordered by increasing preimages. Define an *horizon tree* by starting with the edges of the arrangement to the right of p and begining at $p_1, \ldots, p_n$. At each vertex where two edges intersect, only the upper edge continues. Describe how to compute the horizon tree in $O(n \log n)$ time.

(b) Prove that if there is any arrangement vertex to the right of the path, then there is a *topological triangle* formed by the path and two edges of the horizon tree. A topological triangle is a cycle formed by three edges. Unlike a (geometric triangle, the edges of a topological triangle do not need to be straight.

(c) Prove that no vertex of the arrangement lies inside the uppermost topological triangle.

(d) We can advance the path past the triangle so that the two edges swap order along the path. Describe how to update the horizon tree to become the horizon tree for this modified path.

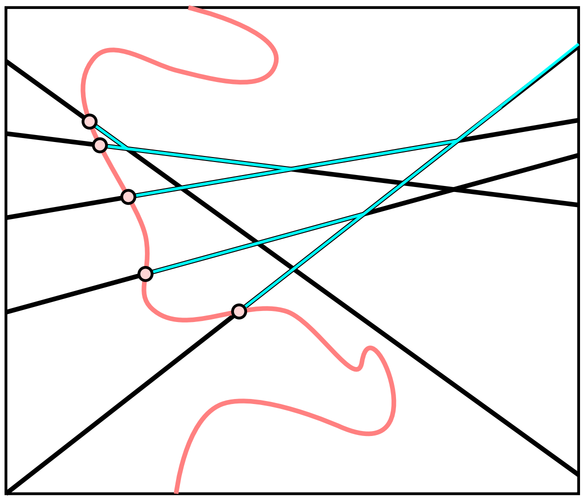

Figure 1: Topological Sweep: the lines of the arrangement are in black, the sweep line is pink, and the horizon tree is cyan.

(e) If we keep triangles in a stack from low to high, we can always have access to the topmost triangle. How do we maintian this stack when we advance past a triangle?

(f) Show that the total time spent mainting the horizon tree is $O(n^2)$.

(30) 5. **Area Of Rectangles**

Suppose that $R_1, \dots, R_n$ is a set of n axis aligned rectangles in the plane. The goal of this problem is to construct an $O(n \log n)$ time algorithm to find the area of their union. There is several important and interesting ideas needed to get such a fast algorithm. We start by using our sweep-line technique. We assume that all the $2n$ vertical(horizontal) sides have distinct values.

(a) More than likely your algorithm from Assignment 1 Problem 6 had $O(n)$ events and required $\Omega(n)$ time per event.

We shall say that a rectangle is live if it intersects our sweep-line. The goal is to come up with a more clever way of representing information about the live rectangles so that we can determine the length of the union of live rectangles in $O(\log n)$ times per event.

Observe that the set of $2n$ vertical rectangle boundaries break our sweep line into at most $2n + 1$ interval. Thus we can make a static binary tree of height $O(\log n)$ with each interval being a leaf, an interval tree. We will not need rotations.

Here is my list of atributes stored at each node. Feel free to pick your own if you can make them work.

- Ply(N) = number of live rectangles containing the interval of N.
- L(N) = the length of the union of live rectangles not including those in Ply(N).

Show how to update these atributes for each of the following operations:

- Insert(R) = add a new live rectangle R.
- Delete(R) = delete a rectangle R.

(b) Use the previous steps to give an algorithm to compute the area of n rectangles in $O(n \log n)$.