CS 274
Computational Geometry (Autumn 2006)
Homework 1

Homework rules: All homeworks in this class are individual assignments and should be treated like take-home exams. You are on your honor to follow these rules:

- Do not seek or obtain assistance from anyone but me.
- Do not help your classmates with the homeworks.
- Do not read any homework solutions or "sample solutions," including those produced by students (except yourself), instructors, or TAs, related to this or any other course on Computational Geometry or a closely related topic. Some of the homework questions I'll use this semester have been used before by me or other teachers at other institutions, and you're on your honor not to consult the solutions.

Giving or receiving hints or answers is grounds for an immediate F in the course.

However, you may consult any books, papers, Web pages, or other inanimate information sources you please, so long as they're not other people's homework solutions or sample solutions. If you use information from such a source, please cite it. I don't believe that the solutions to many of the problems I'll give are available in easy-to-find references, but if you do find one, it's fair game as long as you cite it. (After all, a good literature search should be rewarded.) You may not, however, let your classmates know about your discovery or sources until the deadline has passed.

You are welcome to ask me to clarify anything in the readings, the lectures, or this homework.

I prefer typeset solutions, on paper. I will accept handwritten solutions with the proviso that I reserve the right to give a zero to a solution if there is any word or variable I can't read clearly. (Sorry, I just don't have time to derive from context whether that vague squiggle is an i or a j.) Any such decision is final and irrevocable. By contrast, typeset solutions reserve the right to argue with my grading after the fact.

You may give me your solutions directly, or put them in my CS department mailbox or under my office door (625 Soda) before the deadline. Emailed solutions will be accepted grudgingly, but please use email only if you're out of town or it's otherwise really inconvenient to come to Soda. Snail-mailed solutions (sent to my office; see my Web page for the address) will be accepted if they are postmarked the day before the deadline. Solutions postmarked on the day of the deadline will be accepted if the postmark is clearly from well outside the Bay Area. Late homeworks are not accepted.

Homework 1 is **due at the start of class (1:10 pm) on Wednesday, September 27, 2006**.

[1] Convexity (7 points). Let $P = \{p_1, p_2, \ldots, p_n\}$ be a finite set of points in d-dimensional Euclidean space (E^d). Let q be a point in E^d.

(i) "q is an affine combination of points in P if and only if q' is a linear combination of vectors in V." Give a vector q' and a set V of vectors (defined in terms of q and P) for which this statement is true.

(ii) Prove that if q is an affine combination of points in P, there is a subset $P' \subseteq P$ for which P' contains at most $d+1$ points and q is an affine combination of points in P'. (If you get stuck, go brush up on *matrix rank* in your undergraduate linear algebra textbook.)

(iii) Prove that if q is a convex combination of points in P, there is a subset $P' \subseteq P$ for which P' contains at most $d+1$ points and q is a convex combination of points in P'. (Hint: if there are more than $d+1$ points for which the weight w_i is not zero, you can adjust the weights a bit so that one of them is reduced to zero and the others are still nonnegative but the equalities still hold. If you can't figure out how to prove that, look up the proof of "Radon's Theorem" for a clue.)

[2] Right turn strictly prohibited (4 points). Let P be a set of n points in the plane in general position—that is, no three points in P are collinear.

(i) In pseudocode, write an algorithm that reorders P into a sequence S such that the path that visits the points of S in order (following straight edges from point to point) makes left turns only. In other words, any three consecutive points in S occur in counterclockwise order. Your pseudocode should explicitly give all the orientation tests; e.g. "**if** $\text{ORIENT2D}(p, r, q) > 0$".

(ii) Explain in plain English why your algorithm works.

[3] Maximality (6 points). Let P be a set of n points in the plane. A point $p \in P$ is *maximal* if no point in P is both above and to the right of p.

(i) Give a four-point example P that has both a maximal point not on the convex hull, and a convex hull point that is not maximal.

(ii) For any P, give an $\mathcal{O}(n)$-time algorithm to compute the maximal points of P. Be careful to handle degeneracies correctly (i.e. points with the same x-coordinates or the same y-coordinates).

(iii) Suppose we are restricted to a decision-tree model of computation. Give an $\mathcal{O}(n \log h)$-time algorithm to compute the h maximal points of P. (Adapt Chan's Shattering algorithm.)

[4] Face topology reconstruction (2 points). Suppose we are restricted to a decision-tree model of computation. Practically, this means we cannot use radix sort, bucketing, or other techniques that make k-way decisions for large k; we are only permitted to make binary decisions (i.e. **if** statements). It's well known, for instance, that no comparison-based sort can have a better worst-case running time than $\Theta(n \log n)$, even though radix sort (which is not comparison-based) runs in $\mathcal{O}(n)$ time.
(See http://www.cs.berkeley.edu/∼jrs/61b/lechide/34 if you're not familiar with this kind of lower bound result.)

Suppose we are trying to compute the overlay of two subdivisions, and we have completed the sweepline intersection step. We now have a DCEL with the faces missing, and we want to determine the face topologies next (i.e. match outer boundaries of faces with inner boundaries of their holes). However, we have foolishly not retained any information from the sweepline stage. Explain why the worst-case running time of any algorithm that determines the face topologies is in $\Omega(h \log h)$ (in the decision-tree model), where h is the complexity of the DCEL.

[5] Few turn vertices (3 points). Suppose a polygon has $\mathcal{O}(1)$ turn vertices. Show how to modify the polygon triangulation algorithm (Chapter 3 of the Dutch Book) to run in $\mathcal{O}(n)$ time on this polygon, even in the decision-tree model of computation (e.g. without radix sort).

[6] Red-black closest pair (3 points). Let P and Q be two disjoint point sets in the plane. (Think of them as a red point set and a black point set.) Let $p \in P$ and $q \in Q$ be two points from these sets that minimize the Euclidean distance $|pq|$. Prove that $\overline{pq}$ is an edge of DT$(P \cup Q)$. (This observation leads easily to an $\mathcal{O}(n \log n)$ algorithm for finding p and q, the red-black closest pair.)