6.838: Geometric Computing

Spring 2005
Problem Set 1
Due: Thursday, February 24

Mandatory Part

Problem 1. Width

Let P be a set of n points in the plane. The width of P is defined as

$$\min_{r \text{ of unit length}} \max_{p\in P} r\cdot p - \min_{p\in P} r\cdot p$$

That is, it is equal to the minimum, over all projections of P into a line, of a diameter of the projected set. Yet another interpretation is: squeeze P between two parallel lines; the smallest achievable inter-line distance is precisely the width of P.

Give an $O(n \log n)$-time algorithm that computes the width of P.

Problem 2. Vertical segment intersection

Given n vertical-*only* segments, report all pairs of intersecting segments. For full credit, your algorithm should run in time $O(P + n \log n)$, where P is the number of such pairs.

Problem 3. Textbook, exercise 3.3, p. 60

A *rectilinear polygon* is a simple polygon in which all edges are either horizontal or vertical. Give an example showing that $\lfloor n/4 \rfloor$ cameras are sometimes necessary to guard a rectilinear polygon with n vertices.

Problem 4. Textbook, exercise 4.15, p. 93

A simple polygon P is called *star-shaped* if it containts a point q such that for any $q \in P$, the line segment $p - q$ is contain in P. In other words, p "sees" all points in P, without crossing the boundaries.

Give a (randomized) algorithm to decide whether a simple polygon P is star-shaped. Your algorithm should have expected running time $O(n)$. You can assume P is given by a list of vertices in a clock-wise order.

Optional Theoretical Part

Let S be a set of disks in the plane. The boundaries of the disks are disjoint, but it is possible that one disk D lies entirely inside of another disk D'. In this case, we say that D' *dominates* D.

Give an efficient algorithm which reports the set of all disks which are *not* dominated by any other disk.

Optional Programming Part

Implement a Java applet that constructs a convex hull of a set of (moving) points in the plane. Specifically, the input to your applet is a set S of segments $(p_1, q_1), \ldots, (p_n, q_n)$, as well as number of steps $k \geq 1$ (a segment is represented by a pair of its endpoints). The applet should do the following, for each $i = 0 \ldots k$:

- Compute and draw the set of points S_i, which contains all points $s_i = \frac{i}{k}p + \frac{k-i}{k}q$ for $s = (p, q) \in S$
- Compute and draw the convex hull CH of S_i

In addition: we say that s participates in a *change* in step i, if s_i is a vertex on the boundary of $CH(S_i)$ but not on the boundary of $CH(S_{i-1})$, or vice versa. Your applet should compute the total number of changes per each step i.

Naturally, the applet must also provide a way for the user to specify the input. Graphical specification of the segments (by clicking on the endpoints) is strongly preferred.

Feel free to add any additional bells and whistles.