Assignment 1

The deadline for handing in solutions is Wednesday, 30 January 2013.

Question 1. (20 = 10 + 10 points). We saw several different definitions of convex hull (convex closure) in class. Let P be a finite point set. Prove that the following two definitions are equivalent:

1. The intersection of all convex sets containing P.
2. The set of all convex combinations the points in P.

Question 2. (10 points). Let P be a set of points in the plane. Let $\mathcal{P}$ be the convex polygon whose vertices are points from P and that contains all points in P. Prove that the polygon $\mathcal{P}$ is uniquely defined. (You may assume that the points are in general position).

Question 3. (20 = 5+5+5+5 points). In Lecture 2, we looked at the naive algorithm for computing the convex hull.

(a) One step of the algorithm was to order the edges in the convex hull. Give an optimal algorithm that orders the edges. Your algorithm should take as input an array of edges and return a sorted array of edges.

(b) What is the minimum number and maximum number of edges that can be found on boundary of the convex hull in 2D?

(c) What is the minimum number and maximum number of vertices that can be found on the boundary of the convex hull in 2D?

(d) Analyze the running time of the naive algorithm.

Question 4. (20 = 5 + 5 + 5 + 5 points). The reason we use Oriented Projective Geometry is to create correct, simple functions to evaluate predicates. The first predicate we came across is the side of a flat predicate. Let $p = \langle p_w, p_x, p_y \rangle$, $q = \langle q_w, q_x, q_y \rangle$, and $r = \langle r_w, r_x, r_y \rangle$ be three points given in homogeneous coordinates.

(a) The points p and q define a line $\ell = [w, x, y]$ with normal vector $\langle w, x, y \rangle$. Compute w, x, and y.

(b) Explain how to use the dot product to determine if the point r lies on ℓ, on the same side of ℓ as $\langle w, x, y \rangle$, or on the opposite side of ℓ. Use the following equality in your explanation: $a \cdot b = \|a\| \|b\| \cos\theta$.

(b) Write pseudocode for a function, called sideOfLine, that takes as input a line ℓ (given by 2 homogeneous coordinates) and a point p, both in 2-dimensional homogeneous coordinates and returns 0 if p is on ℓ +1 if p is on the positive side of the plane and -1 if p is on the negative side of the plane. You may assume that there exists a function that computes the determinant of an $n \times n$ matrix.

BONUS [10 points] Generalize the predicate of part (c) to the d-dimensional case. That is, give pseudocode for a function, called sideOfFlat, that takes as input d ordered points in homogeneous coordinates returns 0 if the last point is on the flat defined by the first $d - 1$ points, +1 if it is on the positive side of the flat and -1 if it is on the negative side of the flat. In terms of d, analyze the time complexity of this function.

[Note: Throughout most of this course, we will take for granted that predicates can be evaluated in $O(1)$ time for a fixed dimension. This homework problem is designed to test your understanding of the sideOfFlat predicate. If you wanted to implement code that needs this predicate, we would advise you to use the predicates in established libraries.]

Question 5. (20 = 10 + 10 points). Recall the duality of Oriented Projective Geometry $\langle w, x, y \rangle^* = [w, x, y]$. We use $\langle w, x, y \rangle^*$ to denote the dual of $\langle w, x, y \rangle$.

(a) Find the plane P in $\mathbb{R}^3$ in which the points $\langle 1, x, y_1 \rangle$ and $\langle 1, x, y_2 \rangle$ of the oriented projective plane dualize to parallel lines.

(b) Use the observation from (a) to formulate a dual to the following problem: Given n vertical line segments in the plane, we wish to find a line that stabs, i.e., intersects, all segments. For example, in Figure 1, the red line is a line that stabs all of the green vertical line segments.

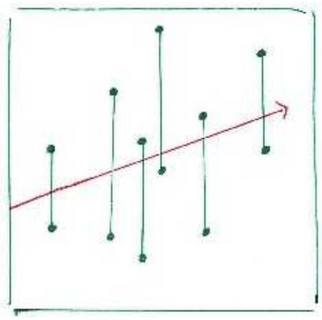

Figure 1: A stabbing line. See qustion 5(b).

Question 6. (15 = 10 + 5 points).

(a) Give pseudocode for a sweepline algorithm that computes the area of the union of a set of n axis aligned rectangles in $O(n^2)$ time. Each rectangle is given as two points: the top right point $t = (t_x, t_y)$ and the bottom left point $b = (b_x, b_y)$.

(b) Show that your algorithm runs in $O(n^2)$ time.

[Note: There does exists an algorithm that computes this in $O(n \log n)$ time.]