Applied Geometric Computing and CGAL — Spring 2005 — Dan Halperin

Assignment no. 2

due: May 4th, 2005

Notice that all the exercises ask for *exact* output; all non-integer input is assumed to be given as rational numbers.
As before, you'll find additional useful information concerning input files, submission of exercises and more in the course's website.

Exercise 2.1: Nearest Jeep over time (30 points)
A group of Jeeps are driving in a flat desert, each in a straight lane and fixed velocity (the lanes or velocities are not necessarily the same, and the lanes may cross). Find the nearest Jeep to the base station over time. Consult the course site for code for computing lower envelopes.
(optional) Give visual evidence (through simulation) to the correctness of your solution.

Exercise 2.2: Intersection of circles (20 points)
In this exercise, the input for each circle is given as three rational numbers: the coordinates of the center, and the radius. Notice that the decision procedures need to be exact, and only the reported coordinates may be approximate.
(a) Read the data of two distinct circles in the plane. Decide whether the circles are tangent, intersect in two points, or are disjoint. If they intersect, print out an approximation of the intersection point(s).
(b) Read the data of three distinct circles in the plane. Decide whether the three circles meet in a single point (YES or NO). If they do, print out an approximation of their common intersection point.
Use algebraic number types (from CORE or LEDA), but otherwise devise and implement your own solution.

Exercise 2.3: Largest common point sets under ε-congruence (50 points)
This is a fairly advanced exercise. Feel free to use as much ready-made software as possible, but be cautious as the required solution should be exact. For example, you can use the maximum cardinality matching in bipartite graphs of LEDA's. (LEDA is available to the school's students.) Any correct solution to this exercise (be it as inefficient as it may) will give you the full credit of 50 points, so you may opt for a brute force solution. However, you are encouraged to devise and implement an efficient solution.
Given two finite sets of points A and B in the plane, we wish to find equally sized subsets $A' \subseteq A$ and $B' \subseteq B$ of maximal cardinality such that points in A' match points in B' under translation, up to distance at most some given ε. Namely, each point a in A' has a unique point b in the translated B' such that the Euclidean distance between a and b is at most ε and A' is the largest cardinality such subset.
The input consists of the coordinates of the points in A and B and the parameter ε. The output should consists of the translation vector, then the size of the matching, followed by pairs of indices of points one from A and one from B. An example of such a pair: if the algorithm finds that the first point read for the set A from the file and the second point read for the set B participate as a pair in the desired matching, the corresponding output will be $(1, 2)$.