EECS 801: Numerical PDEs and Meshing Techniques
Project I: Delaunay Triangulation Project
Due: Thursday, April 7 at 1:00pm

In this project, you will develop a 2D Delaunay mesh generator.

First, implement a two-dimensional triangulation data structure. You may choose which data structure to implement; note that your grade will be based in part by your choice of data structure. Your library should have the ability to add and remove triangles, and to query a triangle's neighbors.

Second, implement an algorithm that constructs two-dimensional Delaunay triangulations. You may choose which Delaunay algorithm to implement, and your grade will be based in part on your choice. See the paper by Su and Drysdale [1] for a comparison of several Delaunay algorithms.

Your code may be written in any language as long as I can read, compile, and execute the code. (Note that I've used C, C++, Fortran, and Matlab, so if you want to choose a different language, please discuss your choice with me first.) I will be running your code on a Linux platform (as opposed to using Windows or a Mac), so please make sure it compiles and runs in such an environment.

Your code should use the same file formats (for the 2D meshes) as the program Triangle (http://www.cs.cmu.edu/~quake/triangle.html). Specifically, it should read a file with the suffix .node, and write a file with the suffix .ele. These file formats are documented on the Triangle web pages. There are two advantages to using these file formats: (1) You can compare your results against those of Triangle, and (2) You can use the Show Me visualization program to view and print your input and output. [Both Triangle and ShowMe are freely available via the web.]

Your code will need to make use of "orientation" and "incircle" tests (whenever possible). If your write your own tests, they'll sometimes give incorrect answers due to roundoff error. My suggestion is that you use exact predicates (that do not suffer from roundoff error). Pointers to such predicates is the following:

- http://www.cs.cmu.edu/~quake/robust.html (floating point inputs by Jonathan Shewchuk)
- http://www-sop.inria.fr/prisme/logiciel/determinant.html (integer inputs by Olivier Deviller)
- your textbook on Geometry and Topology for Mesh Generation (by Herbert Edelsbrunner).

Note that you may need to translate these predicates into the language of your code.

You are welcome to use publicly available libraries or implementations of the following, (so long as none of them was produced by any of your classmates): sorting, selection, binary heaps, balanced search trees, other fundamental non-geometric data structures, command-line switching processing, and geometric primitives like the orientation and incircle predicates. You must write the triangulation and data structure and geometric algorithms all by yourself. However, you may discuss other aspects of the project with up to two other members of the class.

Submission: Submit a gzipped tar file of the code (source code, input files, output files) and ShowMe plots of several of your meshes to me at shontz@ku.edu. In addition, submit a report (on paper) that includes the following:

1. Instructions for compiling and running your code. Be sure to document how to specify the input file, and whether or not there are any options or switches.

2. Timings on random point sets of 10,000, 100,000, and 1,000,000 points. (I will post these point sets as well as some other smaller test sets on the class website soon.) If your code is too slow to finish on the largest point set, just note that. Try to exclude all file I/O from your timings if possible. (If using a timer within your program isn't possible, the Unix time command will do, although file I/O time will be included.)
3. Descriptions of your choices of data structure(s) and algorithm(s) and any extra or unusual feature.

Please limit your write-up to 1-2 pages.

Grading: Your project grade will be based on the following: choice of data structure, choice of algorithm, correctness/accuracy of code, robustness of code, and efficiency, as well as your write-up. This project counts towards 25% of your final grade.

References:

[1] Peter Su and Robert L. Scot Drysdale. *A Comparison of Sequential Delaunay Triangulation Algorithms*. Proceedings of the Eleventh Annual Symposium on Computational Geometry, pages 61-70. Association for Computing Machinery, June 1995.