Don Sheehy
Assistant Professor of Computer Science, University of Connecticut
donald@engr.uconn.edu

Introduction to Computational Geometry

Instructor: Don Sheehy
Email: donald@engr.uconn.edu
Office: ITEB 361
Office Hours: Mon 2-3pm. Wed 10-11am.

See the Course Policies.

Homework

1. Homework 1 on Euclidean Geometry, Predicates, and Convex Hulls
2. Homework 2 on Convexity and Delaunay Triangulation
3. Homework 3: Project Checkpoint

Lecture Notes

1. **What is Computational Geometry (Notes)**

   In this first lecture, I gave a high level overview of the types of problems we will see in the class. Then, we explored some ideas from the computational geometers of antiquity, focusing on Ruler and Compass constructions a la Euclid.

2. **Linear Predicates (Notes)**

   Today, we saw how to generalize the comparison model to geometric problems. We saw that computing the sign of the determinant is usually for answering simple linear questions as well as at least one quadratic question, testing if a point inside a circle.

3. **Convex Hulls (Notes)**

   After starting with a primer on convexity we "discovered" a convex hull algorithm that looked like selection sort. Then, we saw how to improve this algorithm to $O(n \log n)$ by sorting.

4. **Planar Straight-Line Graphs (Notes)**

   Today's lecture was on planar straight line graphs and the basics of topology in the plane.

5. **Data Structures for Planar Graphs (Notes)**

The doubly connected edge list as a way to represent and navigate PSLGs.

6. **Barycentric Decomposition (Notes)**

   The barycentric decomposition as a more general representation of PSLGs and their duals. A brief introduction to polyhedral complexes.

7. **Decompositions and Triangulations (Notes)**

   We start with some more detail on polyhedral complexes and then look at triangulations of polygons.

8. **Intro to Flips (Notes)**

   Today, we went over the homework and then looked at flip operations on a triangulation.

9. **Flipping a triangulation to the Delaunay Triangulation (Notes)**

   This lecture gave the analysis of the termination, correctness, and running time of the GreedyFlip algorithm for computing Delaunay triangulations.

10. **Randomized Incremental Construction of the Delaunay Triangulation (Notes)**

    We proved that the Delaunay triangulation maximizes the minimum angle among all triangulations of a point set. Then we introduced the Randomized Incremental Algorithm for computing the Delaunay triangulation and proved that the expected number of flips is $O(n)$.

11. **Point Location for Randomized Incremental Delaunay Triangulation (Notes)**

    We showed how to do point location for the randomized incremental Delaunay algorithm in $O(n \log n)$ time.

12. **Voronoi Diagrams (Notes)**

    We introduce Voronoi digrams and prove they are dual to Delaunay triangulations.

13. **Projective Duality (Notes)**

    We introduce projective duality both with and without coordinates.

14. **Projective Duality (Notes)**

    We work through some examples of projective duality.

15. **Point location in a planar subdivision (Notes) (More Notes)**

    The history DAG and its analysis.

16. **Half-space Range Counting (Notes)**

    We combine our knowledge of projective duality and planar point location to do half-space range counting.

17. **Orthogonal Range Searching (Notes)**

    We introduce the classic kd-tree and analyze its running time for orthogonal range search queries.

18. **Generic Range Search and Quadtrees (Notes)**

We show how the same basic code used for orthogonal range search can be applied more generally to other types of ranges and other types of hierarchical decomposition. We also introduce the quadtree and compressed quadtree.

19. **Quadtrees for Approximate Nearest Neighbor Search I (Notes)**

    We adapt range searching code to do nearest neighbor search. We then show how shifted quadtrees allow us to get an approximate nearest neighbor.

20. **Quadtrees for Approximate Nearest Neighbor Search II (Notes)**

    We finished the proof that shifted quadtrees guarantees an approximate nearest neighbor. Then, we showed how to simulate the search quickly using space filling curves.

21. **Balanced Quadtrees (Notes)**

    After recapping the different geoemtric search data structures we have seen in the class, we showed a different use of quadtrees related to mesh generation. This led us to study the balanced quadtree and we proved that balancing a quadtree only increases its size by a constant factor.

22. **Linear Programming (Notes)**

    We saw the geometric perspective of linear programming and the linear-time randomized incremental algorithm for solving 2D LPs.

23. **Minimum Enclosing Balls (Notes)**

    We saw a linear-time algorithm for computing the minimum enclosing ball of a set of points in $\R^d$.