

Lecture #1: Wednesday, 5 April 2006
Topics: Course Introduction
Lecturer: Leonidas Guibas

Introduction — Geometric Algorithms

Computational Geometry is now a bit over thirty years old. In the broadest sense, the field is the study of geometric problems from a computational point of view. At its core is a set of techniques for the *design and analysis of geometric algorithms*, certain key geometric data structures, and tools for the robust implementation of these on current computer hardware using familiar computer languages. In what follows we present a few facts about this area and then discuss what we propose to cover in the course.

Some of the excitement of computational geometry is due to a combination of factors: deep connections with classical mathematics and theoretical computer science on the one hand, and many ties with applications on the other. Indeed, the origins of the discipline clearly lie in geometric questions that arose in areas such as computer graphics and solid modeling, computer-aided design, robotics, computer vision, molecular modeling, geography, etc. Not only have these more applied areas been a source of problems and inspiration for computational geometry but, conversely, numerous techniques from computational geometry have been found useful in practice as well. A special challenge in the field is the need to deal with data that has both continuous aspects (point coordinates, plane equations), as well as combinatorial aspects (incidence structures, graphs, polytopes)—and to keep these consistent.

The mix of continuous and discrete representations gives rise to new challenges and indeed the level of mathematical sophistication needed for work in the field has risen sharply in the last decade or so. Nevertheless, many of the new algorithms are simple and practical to implement—it is only their analysis that requires advanced mathematical tools. We intend this course to cover the theoretical, as well as some of the practical, aspects of geometric computation.

Course Outline

There is now so much material in computational geometry that at least a full-year course is needed to cover all the basic techniques. Nevertheless, given some algorithmic preparation, we can cover lots of ground even in one quarter. At least in low dimensions, figures and diagrams can greatly help with intuition. Below is a list of topics to be covered—but we do not promise to cover them in the order listed, and not all may fit in one quarter.

- *Geometric fundamentals*

Computational primitives in two and three dimensions and their implementation; models of computation and lower bounds; geometric duality.

- *Convexity*

Algorithms for convex hulls of point sets in two and three dimensions; convex polygons—properties and algorithms.

- *Arrangements*

The combinatorics of line arrangements, including the zone theorem; sweep-line methods for arrangements—topological sweep; Davenport-Schinzel sequences; many-cell problems.

- *Proximity problems*

Voronoi Diagrams and Delaunay triangulations; algorithms and applications. Approximate Voronoi diagrams.

- *Geometric searching*

Point-location in planar subdivisions; fractional cascading and other efficient data-structuring techniques; three-dimensional analogs. Balanced-aspect-ratio and balanced-box-decomposition trees and their applications.

- *Geometric optimization*

Smallest enclosing balls and ellipsoids, LP-type problems, decimation, parametric search.

- *Triangulations*

Triangulating a simple polygon and applications to shortest-paths; reductions among geometric problems; decompositions of polyhedra; questions of optimality.

- *Visibility and shortest path problems*

Visibility graphs and their uses; Euclidean minimum spanning trees; shortest path problems amidst obstacles.

- *Geometric sampling techniques*

Random sampling for partitioning; randomized incremental algorithms; ϵ -nets; making randomized algorithms deterministic; cuttings and their applications. Core sets and applications.

- *Partition trees and range searching*

The ham-sandwich theorem; decimation methods; range-searching problems of various kinds.

- *Robustness in geometric computation*

Issues in topological consistency; handling of degeneracies; numerical evaluation of geometric primitives; rounding of geometric structures; robust algorithms.

Bibliography

The main text for the course is the course reader, comprised of a number of survey articles plus lecture notes written by the lecturer or scribed with the help of students in previous years of this class.

There are now several excellent new textbooks commercially available that cover most of the same material as well. A very well written elementary introduction is *Computational Geometry in C* by J. O'Rourke (Cambridge U. Press, second edition, 1998). The book *Computational Geometry, Algorithms and Applications*, by M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf (Springer Verlag, 1997) has an excellent selection of topics, each well motivated by a practical application. The newest book in English *Algorithmic Geometry*, by J-D. Boissonnat and M. Yvinec (Cambridge U. Press, 1998, translated from the French by H. Brönnimann) offers more advanced coverage at the level of a research monograph. An extensive survey of the field is given in the *Handbook of Discrete and Computational Geometry*, edited by J. Goodman and J. O'Rourke (CRC Press, 2004—second edition), and in the *Handbook of Computational Geometry*, edited by J. Sack and J. Urrutia (North-Holland, 2000).

The previous generation of books includes the text by K. Mulmuley *Computational Geometry: an Introduction through Randomized Algorithms* (Prentice Hall, 1993). Also in that group are the excellent monograph *Davenport-Schinzel Sequences and their Geometric Applications* (Cambridge U. Press, 1995) by M. Sharir and P. Agarwal, as well as the survey text *Combinatorial Geometry* by J. Pach and P. Agarwal (Wiley-InterScience, 1995). Currently available older bibliography includes three books. The oldest one is volume III of K. Mehlhorn's *Data Structures and Algorithms*, titled *Multidimensional Searching and Computational Geometry* (Springer-Verlag, 1984). Next is the classic *Computational Geometry, An Introduction*, by F. P. Preparata and M. I. Shamos (Springer-Verlag, 1985), and most recent is *Algorithms in Combinatorial Geometry* by H. Edelsbrunner (Springer-Verlag, 1987).

Papers in computational geometry appear in a variety of computer science journals, including the *ACM Transactions on Graphics*, *Algorithmica*, the *Journal of Algorithms*, the *Journal of the ACM*, the *SIAM Journal on Computing*, and others. There is a twenty-year-old journal, *Discrete and Computational Geometry*, that is primarily devoted to this field. Two more such journals began publication some years back, the *International Journal of Computational Geometry & Applications*, and *Computational Geometry, Theory and Applications*.

There is an annual conference, the *ACM Annual Conference on Computational Geometry*, now in its twenty-second year, whose proceedings are a useful reference for much of the work in the area. There is also a Canadian conference in the field that started

fifteen or so years ago, as well as annual workshops in Europe and the Far East. Other well established theory conferences, such as *STOC*, *FOCS*, *SODA*, or *WADS/SWAT*, also get a good share of high quality geometry papers—the last two usually contain a substantial fraction.

The survey papers included in the primary reader contain as a whole a reasonable set of references to the main research papers in the field. An on-line geometry paper data-base containing several thousand papers is also available and will be described more fully in the web page for the course. More bibliographic data can be found on-line at <http://compgeom.cs.uiuc.edu/~jeffe/compgeom/biblios.html>

Office hours, address data, etc.

- | | |
|--------------------|---------------------------------|
| Instructor: | Prof. Leonidas J. Guibas |
| office: | Clark S293 |
| tel.: | 723-0304 |
| e-mail: | guibas@cs.stanford.edu |
| office hours: | Tuesday, 1:00 – 2:30 pm |

- | | |
|----------------------------|---|
| Teaching Assistant: | Nikola Milosavljevic |
| office: | Clark S257 |
| tel.: | 725-6532 |
| e-mail: | nikolam@cs.stanford.edu |
| office hours: | Thursday, 11:00 am – 12:00 noon and
Friday, 12:00 noon – 2:00 pm |

- | | |
|--------------------------|---------------------------|
| Course Secretary: | Hoa Nguyen |
| office: | Gates 146 |
| tel.: | 723-4137 |
| e-mail: | hoa@robotics.stanford.edu |

Handouts and Course Notes

Every student should have a copy of the primary course reader. There may be additional handouts during the course. When we cover material that is not well represented in the text or the reader, the instructor may call for student volunteers to help in scribing the corresponding lecture(s).

If you do not already have experience, we recommend that you become familiar with L^AT_EX and some drawing program that produces postscript files, such as Xfig on Unix workstations, or Adobe Illustrator on a PC or Mac. We will provide you with L^AT_EX style files for the course notes, and macros for incorporating any figures you produce. Handout #3 in the reader discusses this process in more detail.

Scribes should provide a draft of the lecture notes to the instructor, in both hardcopy and electronic form, no later than a week after the lecture being scribed.

Web page

The most up-to-date information on the class is available at

<http://graphics.stanford.edu/courses/cs368-06-spring/> or
<http://cs368.stanford.edu/>

This URL contains an evolving syllabus, and copies of handouts and homeworks, as well as links to useful resources on the web.

Homeworks, Exams, Grading, etc.

The course will have three substantial homework assignments. There will be no final exam, but there will be a midterm whose function will be to test breadth but not depth. The schedule will be as follows:

homework	handed out	due
#1	Monday, 17 April	Monday, 1 May
#2	Monday, 1 May	Monday, 15 May
#3	Monday, 15 May	Monday, 5 June
midterm	in class	Wednesday, 10 May

Students may elect to take this class through either a *theoretical track* or through an *applied track*. Students in the theoretical track will have to do sets of ‘paper-and-pencil’ type problems as part of each homework. Those electing the applied track will substitute a programming project in lieu of some of the problems. Every student taking the class for credit is expected to do all the homeworks (according to the track they are in) and take the midterm. Switching among tracks is permitted only up to the first homework due date. The programming track will require a project; all programming projects will be demonstrated to the entire class at the end of the quarter.

Collaboration with other students in the class in doing the homeworks is permitted in groups of up to three students, in fact collaboration is encouraged. However, each write-up must be individually composed and the names of the collaborators must be listed for each problem. Those in the applied track may work in groups of up to three students for the programming project(s). For the final grade we will count each of the homeworks and the midterm as 25% of the grade. *Please do the homework*—there is no other way to learn the material.

It is very important in this course that every homework be turned in on time. We recognize that occasionally there are circumstances beyond one’s control that prevent an assignment from being completed before it is due. You will be allowed two classes of grace during the quarter. This means that you can either hand-in two assignments late by one class, or one assignment late by one week. Any other assignment handed in late will be penalized by 20% for each class that it is late, unless special arrangements have been made previously with the instructor.

All course work must be handed in by Wednesday, 7 June, 2006.

Homework solutions will be handed out in hardcopy in class. All other handouts and class materials will be available on the web.
