Lecture I
GEOMETRIC MODELING LECTURE NOTES

Geometry is everywhere. Most basic geometric objects: points and lines. We owe much of our intuitions about geometry to the development by the Greeks. In Mathematics, geometry is one of the most important sources of mathematical intuitions, profoundly affecting practically every branch of mathematics. Even in number theory, there is an area called the geometry of numbers.

§1. What is Geometric Computing?

What is geometry? Geometry is about geometric relations that hold in a collection (arrangement) of geometric objects. From the continuous Cartesian plane, we extract discrete (combinatorial) relations which we identify as "geometric". For instance, the relation of "point on a line", "point left of directed line", or "3 collinear points", "2 parallel lines". The study of such relations is the beginning of geometry.

Historical Perspective. (A) Euclid, Descartes, Hilbert.

(B) Combinatorial Vein: Erdos

(C) Complexity Vein: since the 1970s.

What is Geometric Modeling? In fields where geometry is a significant component, the issues of geometric modeling arises. In various engineering disciplines and architecture, large geometric models are needed. The canonical example is the field of CAD/CAM. In this course, we focus on two areas:

(A) Solid and surface modeling.

(B) GIS Applications.

Visualization. We should add that part and parcel with modeling is the problem of "visualization". This subject is developing into a discipline all of its own, somewhat distinct from the modeling problem. In geometric modeling, we want to provide primitives that allows the user to design the desired model. In visualization, we begin with the model, and want to visualize it. Visualizing means providing the user with tools for "interacting" with the model in order to general various views. This becomes a highly non-trivial problem when the models becomes large, or when the data is high dimensional (e.g., time-varying volumetric data). In this course, we shall be as interested in the modeling as with the visualization.

§2. What is GIS?

Today, we are familiar with map servers on the internet. It is a simple example of geographical information system.

There is no simple definition for the field of GIS. Naively, GIS is about the application of computer technology to the age-old problem of map making (cartography) and their presentations. Already in this simplistic starting point, we can identify several deeper issues:

- Map presentations in a map is not a straightforward problem, but there are implicit models and conventions that constrain presentation. For instance, the simplistic idea that map entities are correctly scaled breaks down for the width of roads or rivers – on any medium, whether in print or on a computer screen, there is a lower limit on how fine a feature can be represented.
- Not everything that is known about a geographic terrain can be presented, so selective choices must be made. Traditional cartography has map themes to separate out the various elements present in a map. But with computer technology, much greater possibilities present themselves. Indeed, one of the biggest advantages of computerization is that we can now present highly customized representations, suited to individual needs.
- GIS is concerned with a much larger range of data than just maps. We might say that it is concerned with any kind of "geo-referenced" data sets. Such data sets has many sources and comes in many forms. They may be multispectral satellite images of the earth, data above the land and water surface, data below land or water surface, etc.
- GIS is not only concerned about processing and presenting images of such data sets, but its functions include analysis, modeling and planning.

See [Oostrom, p.3] for several definitions of the GIS field. We saw that the application of computer technology has changed the nature of cartography itself.

GIS applications is growing rapidly, and is a multibillion dollar industry. Its technology are of interest to government, the military, citizen groups, businesses as well as individual citizens.

GIS as a computational issue. We are interested in GIS as a computational problem. An interesting view comes from databases: think of the data set(s) as a database system to which we direct queries. Let us also assume that the query output should always be an geo-referenced image. For instance, if we ask "where is the the point (x, y)?" we expect a map in which the point (x, y) is indicated by some means (e.g., a push pin).

A closely related view comes from visualization. In visualization, the "navigational paradigm" is a very powerful one. The whole data set (map) is very large, and we can either view the whole map in low resolution or only view parts of it in higher resolution. The goal of the GIS system is to provide such views in real time. The typical navigational controls to achieve this is the zoom and pan paradigm.

Both views demand that we represent (i.e., model) the geo-spatial data in a form that is conducive to the types of queries to be supported. That is where computer science comes in – computer science has many data structures and algorithms that can support this function.

§3. What is Computational Geometry?

It is the application of algorithmic techniques in the study of geometric objects and their computation. The field of computational geometry is part of the larger subject of algorithmics. To give an idea of this field, we can classify some key problems studied in the name of computational geometry:

1. Construction of geometric objects. Examples: Voronoi diagrams, convex hulls, arrangements of cells defined by polynomials.

2. Searching for an object with specific properties, within a class of implicitly given objects. Let us clarify this with an example: finding a shortest path between two points p, q amidst a set of polygonal objects is of this form: the implicit class is the set of all paths from p to q, and the specific property is the minimum length of the path. Most optimization problems (including linear programming) falls into this category. Examples: Constructing the largest (area,perimeter) k-gon inscribing (circumscribing) a given n-gon.

3. Preprocessing problems. That is, given a large data set S, we want to preprocess it into some efficient data structure $D(S)$ which can support certain classes of queries on S. For instance, if S is a set of points, and we want to support queries of the form: given a query point q, find the point in S that is closest to q. One data structure to support this problem is a point location data structure for the Voronoi diagram of S. Another example is the range query problem.

4. Decomposition of objects. Given a subset of Euclidean space, we want "decompose" it into a collection of subsets which either covers the set or is a partition, subject to restrictions. Generally speaking, covering problems are harder than partition problems. Example: given a polygon, we want to cover it with triangles or partition it into triangles. Perhaps we want to minimize the number of triangles. The latter is the problem of triangulation of polygons with an interesting history.

5. Computing functionals on geometric objects. Example: computing the measure (volume) of a subset in Euclidean space. If the subset is presented as the union of a set of boxes, this is the measure problem of Klee. In GIS applications, we often have to compute the area of some specified geographical region.

6. Intersection problems. Example: Given some objects, do some specific types of intersections among these objects occur? Example: Helly-like properties, classical theorems of Carathéodory, Steinitz, Radon, ham-sandwich theorems, transversaal theory.

7. Convexity: there is a book by Grötschel,Lovász and Schrijver on this fascinating topic.

8. Motion Planning problems. Originally motivated by robotics, the basic problem is that of computing some path in configuration space, which is implicitly given by a set of geometric objects.

§4. Preliminary

Some basic notations. We use N, Z, R, C to denote the set of natural numbers, the integers, the reals and the complex numbers. Some basic notions of point set topology is assumed.

Our geometry will be exclusively in Euclidean space R^n for some natural number $n \geq 1$. Many problems of interest appear in $n = 2$ or 3. In R^n, the n coordinate axes are labeled by $x_1, \ldots, x_n$. Using a standard notational device in computer programming languages, if $p \in R^n$ is a point, we will write $p.x_i$ to denote the i-th coordinate of p ($i = 1, \ldots, n$). When $n \leq 3$, we may write x, y, z for x_1, x_2, x_3, and $p.z$ instead of $p.x_3$, etc.

The basic concepts of linear algebra and linear spaces is assumed. Linear algebra is concerned with the solution of linear equations. If $\ell(x_1, \ldots, x_n) = a_0 + \sum_{i=1}^n a_i x_i$ is a linear function ($a_i \in R$), then the set

$$\{(r_1, \ldots, r_n) \in R^n : \ell(r_1, \ldots, r_n) = 0\}$$

is a **hyperplane**. An **affine subspace** of R^n is obtained as a finite intersection of hyperplanes. In fact, each affine subspace S can be obtained as the intersection of i hyperplanes for some $i = 1, \ldots, n$. If i is minimum, then S may be identified with R^{n-i} and $n - i$ is the **dimension** of S. An affine subspace of dimension 1 is called a line, while a hyperspace is dimension $n - 1$.

We will also need "semi-linear algebra" which addresses linear inequalities. The set

$$\{(r_1, \ldots, r_n) \in R^n : \ell(r_1, \ldots, r_n) \geq 0\}$$

is closed **half-space**. The intersection of half-spaces yields convex polyhedrons. The linear programming problem is a typical problem in semi-linear algebra.

References