# Representing Geometry in Computer Graphics

Rick Skarbez, Instructor
COMP 575
September 18, 2007

# Adam Powers

Richard Taylor
Information International, Inc. (III), 1981

# Announcements

- Programming Assignment 1 is out today
  - Due next Thursday by 11:59pm
- ACM Programming Contest

# ACM Programming Contest

- First regional competition (at Duke) is Oct 27
- Top teams at regionals get a free trip to worlds
  - This year, in scenic Alberta Canada
- First meeting is this Thursday (09/20) at 7pm in 011
- All are welcome!

# Last Time

- Introduced the basics of OpenGL
- Did an interactive demo of creating an OpenGL/GLUT program

# Today

- Review of Homework 1
- Discussion and demo of Programming Assignment 1
- Discussion of geometric representations for computer graphics

# Homework 1

- Noticed 2 recurring problems
  - Normalized vs. Unnormalized points and vectors
  - Matrix Multiplication

# Points and Vectors in Homogeneous Coordinates

- To represent a point or vector in n-D, we use an (n+1)-D (math) vector
  - Add a ‘w’ term
- We do this so that transforms can be represented with a single matrix

# Points and Vectors in Homogeneous Coordinates

- Vectors (geometric) represent only direction and magnitude
  - They do not have a location, so they cannot be affected by translation
  - Therefore, for vectors, w = 0
- Points have a location, and so are affected by translations
  - Therefore, for points, w != 0

# Points and Vectors in Homogeneous Coordinates

- A “normalized” point in homogeneous coordinates is a point with w = 1
  - Can normalize a point by dividing all terms by w:

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \longrightarrow \begin{bmatrix} \frac{x}{w} \\ \frac{y}{w} \\ \frac{w}{w} \end{bmatrix}$$

# Points and Vectors in Homogeneous Coordinates

- A “normalized” vector is a vector with magnitude = 1
  - Can normalize a vector by dividing all terms by the vector magnitude = sqrt(x^2 + y^2 + z^2) (in 3-D)

# Matrix Multiplication

- Only well defined if the number of columns of the first matrix and the number of rows of the second matrix are the same
  - Matrix * Matrix = Matrix
  - *i.e.* if F is m x n, and G is n x p, then FG if m x p

$$(\mathbf{FG})_{ij} = \sum_{k=1}^{m} F_{ik} G_{kj}$$

## Programming Assignment 1

- Demo

## Geometric Representations in Computer Graphics

- There are 2 most common ways of representing objects in computer graphics
  - As procedures
    - *i.e.* splines, constructive solid geometry
  - As tessellated polygons
    - *i.e.* a whole bunch of triangles

## Analytic Representations

- Advantages:
  - Can be as precise as you want
    - That is, can compute as many points on the function as you want when rendering
  - Derivatives, gradients, etc. are directly available
    - Continuous and smooth
  - Easy to make sure there are no holes

## Analytic Representations

- Disadvantages:
  - Not generalizable
    - Therefore, SLOW!
  - Often complicated

## Tessellated Polygons

- Advantages:
  - Very predictable
    - Just pass in a list of vertices
    - Therefore, can be made VERY FAST

## Tessellated Polygons

- Disdvantages:
  - Only know what the surface looks like at the vertices
  - Derivatives, gradients, etc. are not smooth across polygons
  - Can have holes

# Analytic Representations

- Probably the most obvious way to use analytic representations is to just represent simple objects as functions
  - Planes: $a\mathbf{x} + b\mathbf{y} + c\mathbf{z} + d = 0$
  - Spheres: $(\mathbf{x} - x_o)^2 + (\mathbf{y} - y_o)^2 + (\mathbf{z} - z_o)^2 = r^2$
  - etc.
    - We'll do this when we're ray-tracing

# Analytic Representations

- There are many more procedural methods for generating curves and surfaces
  - Bézier curves
  - B-Splines
  - Non-rational Uniform B-Splines (NURBS)
  - Subdivision surfaces
  - etc.
- We may cover some of these later on

# Computer-Aided Design

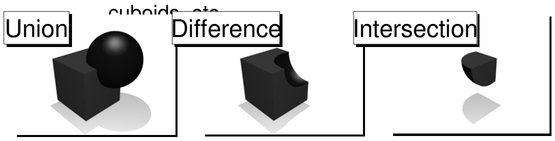
- An application where analytic representations are commonly used is Computer-Aided Design (CAD)
  - Why?
    - Accuracy is very important
    - Speed is not that important

# Solid Modeling

- Another reason analytic models are often used in CAD is that they can directly represent solids
  - The sign of a function can determine whether a point is inside or outside of an object

# Constructive Solid Geometry (CSG)

- A method for constructing solid objects from Boolean combinations of solid primitives
  - Operators: union, difference, intersection
  - Primitives: spheres, cones, cylinders, cuboids, etc.




# Why Triangles?

- So, why are triangles the primitives of choice, and not squares, or septagons?
  - Simplest polygon (only 3 sides)
    - Therefore smallest representation
  - Guaranteed to be convex
  - Easy to compute *barycentric coordinates*

## Barycentric Coordinates

- Every triangle defines a (possibly) non-orthogonal coordinate system
  - Let **a**, **b**, and **c** be the vertices of the triangle
  - Arbitrarily choose **a** as the origin of our new coordinate system
  - Now any point **p** can be represented
    $\mathbf{p} = \mathbf{a} + \beta(\mathbf{b} - \mathbf{a}) + \gamma(\mathbf{c} - \mathbf{a})$
    or, with $\alpha = 1 - \beta - \gamma$, as
    $\mathbf{p} = \alpha\mathbf{a} + \beta\mathbf{b} + \gamma\mathbf{c}$

## Barycentric Coordinates

- Why are these so great?
  - Easy to tell if a point is inside the triangle
    - A point is inside the triangle if and only if $\alpha$, $\beta$, and $\gamma$ are all between 0 and 1
  - Interpolation is very easy
    - Needed for shading

## Next Time

- Beginning our discussion of lighting and shading