Overview

- 3D solid model representations
  - Implicit models
  - Super/quadsrics
  - Blobsies
  - Swept objects
  - Boundary representations
  - Spatial enumerations
  - Distance fields
  - Quadrees/octrees
  - Stochastic models

Implicit Solid Modeling

- Idea:
  - Represents solid as the set of points where an implicit global function takes on certain values
  - Usually
    - $F(x,y,z) < 0$, points inside of object
    - $F(x,y,z) = 0$, points on object’s surface
    - $F(x,y,z) > 0$, points outside of object
  - Primitive solids are combined using CSG
  - Composition operations are implemented by functionals which provide an implicit function for the resulting solid

Quadratic Surfaces

- Sphere
  - $x^2 + y^2 + z^2 = r^2$

Superellipsoid Surfaces

- Generalization of ellipsoid
- Control parameters $s_1$ and $s_2$
  - $\left( \frac{x}{r_1} \right)^{\frac{1}{s_1}} + \left( \frac{y}{r_2} \right)^{\frac{1}{s_2}} = 1$
  - If $s_1 = s_2 = 1$ then regular ellipsoid
  - Has an implicit and a parametric form!

Superellipsoid Surfaces

The general superellipsoid has a parametric representation in terms of surface parameters $-1 \leq r < \infty$, $n < u < n$

$$x(u,v) = Ac(v, s) c(u, s)$$
$$y(u,v) = Bc(v, s) s(u, s)$$
$$z(u,v) = C\cos(v, s)$$

where the auxiliary functions are
$$c(\omega, m) = \text{sgn}(\cos\omega) \cos\omega^m$$
$$s(\omega, m) = \text{sgn}(\sin\omega) \sin\omega^m$$
and the sign function $\text{sgn}(x)$ is
$$\text{sgn}(x) = \begin{cases} -1, & x < 0 \\ 0, & x = 0 \\ +1, & x > 0. \end{cases}$$
Superellipsoid Surfaces

- Normals defined by
  
  \[
  n_x(u,v) = (1/A)c(v, 2-s_1)c(u, 2-s_2) \\
  n_y(u,v) = (1/B)c(v, 2-s_1)s(u, 2-s_2) \\
  n_z(u,v) = (1/C)s(v, 2-s_1)
  \]

- A, B and C are scale factors of the X, Y & Z coordinates
- \(s_1\) is the shape parameter for longitude lines
- \(s_2\) is the shape parameter for latitude lines

CSG with Superquadrics

Blobby Objects

- Do not maintain shape, topology
  - Water drops
  - Molecules
  - Force fields
- But can maintain other properties, like volume

Gaussian Bumps

- Model object as a sum of Gaussian bumps/blobs
  \[
  f(x,y,z) = \sum b_k e^{-\frac{r_k}{T}} - T = 0
  \]
- Where \(r_k^2 = x_k^2 + y_k^2 + z_k^2\) and \(T\) is a threshold.
Metaballs (Blinn Blobbies)

Ray-traced Metaballs

Implicit Modeling System
U. of Calgary

• Combine "primitives"
  – Points, lines, planes, polygons, cylinders, ellipsoids
• Calculate field around primitives
• View iso-surface of implicit function

Implicit Modeling System
U. of Calgary

Can apply blends and warps

Sweep Representations

• An alternative way to represent a 3D object
• Idea
  – Given a primitive (e.g. polygon, sphere)
  – And a sweep (e.g. vector, curve...)
  – Define solid as space swept out by primitive

Sweep Representations

• Issues:
  – How to generate resulting surface?
  – What about self-intersections?
  – How to define intersection?
Approximate Representations

- Idea: discretize the world!
- Surface Models
  - Mesh, facet and polygon representations
- Volume Models
  - spatial enumeration
  - voxelization

Examples

- From exact to facets…

Boundary Representation
Solid Modeling

- The de facto standard for CAD since ~1987
  - BReps integrated into CAGD surfaces + analytic surfaces + boolean modeling
- Models are defined by their boundaries
- Topological and geometric integrity constraints are enforced for the boundaries
  - Faces meet at shared edges, vertices are shared, etc.

Let’s Start Simple:
Polyhedral Solid Modeling

- Definition
  - Solid bounded by polygons whose edges are each a member of an even number of polygons
  - A 2-manifold: edges members of 2 polygons

Properties of 2-Manifolds

- For any point on the boundary, its neighborhood is a topological 2D disk
- If not a 2-manifold, neighborhood not a disk

Euler’s Formula

- For simple polyhedron (no holes):
  \#Vertices - \#Edges + \#Faces = 2
- If formula is true the surface is closed
**Euler’s Formula (Generalized)**

\[ V - E + F - H = 2(C - G) \]

- Genus is the # holes through the object
- Euler Operators have been the basis of several modeling systems (Mantyla et al.)

**Euler Operators**

<table>
<thead>
<tr>
<th>Operator Name</th>
<th>Meaning</th>
<th>V</th>
<th>E</th>
<th>F</th>
<th>L</th>
<th>S</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>MEV</td>
<td>Make an edge and a vertex</td>
<td>+1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>MFE</td>
<td>Make a face and an edge</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>MSFV</td>
<td>Make a shell, a face and a vertex</td>
<td>-1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>MSG</td>
<td>Make a shell and a hole</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>MEL</td>
<td>Make an edge and a loop</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Lim L → H, Shell S → C

**Steps to Creating a Polyhedral Solid Modeler**

- **Representation**
  - Points, Lines/Edges, Polygons
- **Modeling**
  - Generalization of 3D clipping to non-convex polyhedra, enables implementation of booleans

**State of the Art: BRep Solid Modeling**

- … but much more than polyhedra
- Two main (commercial) alternatives
  - All NURBS, all the time
    - Pro/E, SDRC, …
  - Analytic surfaces + parametric surfaces + NURBS + …, all stitched together at edges
    - Parasolid, ACIS, …

**Issues in Boundary Representation Solid Modeling**

- Very complex data structures
- Complex algorithms
  - Manipulation, booleans, collision detection
- Robustness
- Integrity
- Translation
- Features
- Constraints and Parametrics

**Other Issues: Non-Manifold Solids**

- There are cases where you may need to model entities that are not entirely 3D
Cell Decomposition

- Set of primitive cells
- Parameterized
- Often curved
- Compose complex objects by gluing cells together
- Used in finite-element analysis

Spatial Occupancy Enumeration

- Brute force
  - A grid
- Pixels
  - Picture elements
- Voxels
  - Volume elements
- Quadtrees
  - 2D adaptive representation
- Octrees
  - 3D adaptive representation
  - Extension of quadtrees

Brute Force Spatial Occupancy Enumeration

- Impose a 2D/3D grid
  - Like graph paper or sugar cubes
- Identify occupied cells
- Problems
  - High fidelity requires many cells
- “Modified”
  - Partial occupancy

Distance Volume

- Store signed distance to surface at each voxel

Quadtrees

- Hierarchically represent spatial occupancy
- Tree with four regions
  - NE, NW, SE, SW
  - “dark” if occupied
Quadtree Data Structure

Octree

Boolean Operations on Octrees

Adaptive Distance Fields

Applications for Spatial Occupancy Enumeration

Issues with Spatial Occupancy Enumeration

- Many different applications
  - GIS
  - Medical
  - Engineering Simulation
  - Volume Rendering
  - Video Gaming
  - Approximating real-world data
- Approximate
  - Kind of like faceting a surface, discretizing 3D space
  - Operationally, the combinatorics (as opposed to the numerics) can be challenging
  - Not as good for applications wanting exact computation (e.g. tool path programming)
Binary Space Partition Trees (BSP Trees)

- Recursively divide space into subspaces
- Arbitrary orientation and position of planes
- Homogeneous regions are leaves called in/out cells

Statistical Representations

- Store density (material vs. void)
- Statistical description of geometry
- Goal – describe the porosity without storing the geometry information

Stochastic Geometry

- Need some way of converting a solid into some representative statistical form
- From each material voxel, calculate the distance to the nearest voxel that is not material
- Repeat for void voxels
- Store distributions:
  - one for empty space
  - one for material
  - density value

Application: Biological Models

- Bone tissue
- MRI data
- Other biological data
- Solid modeling

Application: Surface Texture
Application: Surface Texture

Programming Assignment 4

- Implement parametric form of superellipsoids
- Iterate through u and v parameters
- Calculate point and normal for each (u,v) pair
- Only calculate one point at each of the poles
- Top and bottom rows should be a triangle fan with poles at center
- Other rows are quads that are broken into triangles
- Output mesh as Open Inventor

End