Solid Modeling
Foley & Van Dam, Chapter 11.1 and Chapter 12
Solid Modeling

- Polygon Meshes
- Plane Equation and Normal Vectors
- Volume Representation
  - Sweep Volume
  - Spatial Occupancy Enumeration
  - Binary Space Partition Tree
  - Constructive Solid Geometry
  - Boundary Representation
Polygon Meshes

A polygon mesh is a collection of polygons, along with a normal vector associated to each polygon vertex:

- An edge connects two vertices
- A polygon is a closed sequence of edges
- An edge can be shared by two adjacent polygons
- A vertex is shared by at least two edges
- A normal vector pointing “outside” is associated with each polygon vertex
Polygon Meshes

Properties:

- **Connectedness**: A mesh is connected if there is a path of edges between any two vertices.
- **Simplicity**: A mesh is simple if the mesh has no holes in it.
- **Planarity**: A mesh is planar if every face of it is a planar polygon.
- **Convexity**: The mesh is convex if the line connecting any two points in the mesh belongs to the mesh.
Representing Polygon Meshes

- **Explicit (vertex list)**
  
  $P_1 = (V_1, V_2, V_4)
  
  $P_2 = (V_2, V_3, V_4)$

- **Pointers to a vertex list**
  
  $V = (V_1, V_2, V_3, V_4)$
  
  $P_1 = (1, 2, 4)$
  
  $P_2 = (4, 2, 3)$

- **Pointers to an edge list**
  
  $V = (V_1, V_2, V_3, V_4)$
  
  $E_1 = (1, 2, P_1, \lambda)$  \( \lambda \) Represents null
  
  $E_2 = (2, 3, P_2, \lambda)$
  
  $E_3 = (3, 4, P_2, \lambda)$
  
  $E_4 = (4, 2, P_1, P_2)$
  
  $E_5 = (4, 1, P_1, \lambda)$
  
  $P_1 = (E_1, E_4, E_5)$
  
  $P_2 = (E_2, E_3, E_4)$
Representing Polygon Meshes

• Explicit
  Space effective for single polygons
  Not very informative

• Pointers to Vertex List
  Saves spaces for shared vertices
  Easy to modify a single vertex
  Difficult to find polygons sharing an edge
  Multiple drawing and clipping of shared edges

• Pointers to Edge List
  Solves multiple drawing and clipping problem
  Still hard to find all edges sharing a vertex
Plane Equations

\[ Ax + By + Cz + D = 0 \quad \text{or} \quad \begin{bmatrix} A & B & C & D \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = 0 \]

• \( N=[A,B,C] \) is the plane normal (\(|N|=1\) )
• \( D= -P*N \)
• \(|D| \) is the plane distance from the origin.
• \( N \) points “outside”, so:
  • \( Ax+By+Cz+D<0 \), \((x,y,z)\) is “inside”
  • \( Ax+By+Cz+D>0 \), \((x,y,z)\) is “outside”
• Three non collinear points \( V_1, V_2, V_3 \) are sufficient to find the coefficients \( A, B, C \) and \( D \)
Plane Equations

- Given $V_1, V_2, V_3$, the plane normal (or the coefficients $A, B$ and $C$) can be computed with the cross product: $S = (V_3 - V_1) \times (V_2 - V_1)$ and $N = S / |S|$
- $D$ can be found by plugging any point of the plane in the equation $Ax + By + Cz + D = 0$
- The plane equation is not unique
Plane Equations

• Example:
  Find the equation of the plane containing the points 
  \(V_1=(2,1,2), \ V_2=(3,2,1)\) and \(V_3=(1,1,3)\)

First we compute the vectors

\[V_2-V_1 = (1,1,-1)\text{ and } V_3-V_1= (-1,0,1)\]

The A, B and C coefficients of the equation are given by
the cross product \(S=(V_2-V_1) \times (V_3-V_1)\)

\[N = [A, B, C] = S/|S| = (1,0,1)/|(1,0,1)| = \left(\frac{1}{\sqrt{2}}, 0, \frac{1}{\sqrt{2}}\right)\]

Finally we substitute \(V_1\) in the equation to find \(D\).

To simplify the computation, we use \([A,B,C]=(1,0,1)\)

\[Ax+By+Cz+D=0\]

\[1*2+0*1+1*2+D=0\]

\[D=-4\]

And the plane equation is \(x+z-4=0\)

The equation with the unitary normal is \(\frac{1}{\sqrt{2}}\left(x + z - 4\right) = 0\)
Volume Representation

- A collection of techniques to represent and define volumetric objects
- Desired properties:
  - Rich representation
  - Unambiguous
  - Unique representation
  - Accurate
  - Compact
  - Efficient
  - Possible to test validity
Volume Representation

- Volume Representation
  - Primitive Instancing
  - Sweep volumes
  - Spatial Occupancy (voxels, Octree, BSP)
  - Constructive Solid Geometry
- Boundary Rep.
  - Polyhedra
  - Free form
Primitive Instancing

• Define a family of parameterized objects
• The definition is procedural (a routine defines it)
• Not general, must be individually defined for each family of objects

Example: Wheels/Gears

Diameter = 10
Teeth = 16
Hole = 3

Diameter = 8
Teeth = 24
Hole = 5
Sweep Volume

- **Sweep Volume**: sweeping a 2D area along a trajectory creates a new 3D object

- **Translational Sweep**: 2D area swept along a linear trajectory normal to the 2D plane

- **Tapered Sweep**: scale area while sweeping

- **Slanted Sweep**: trajectory is not normal to the 2D plane

- **Rotational Sweep**: 2D area is rotated about an axis

- **General Sweep**: object swept along any trajectory and transformed along the sweep
Sweep Volume

Translational and rotational sweep volumes
Spatial Occupancy Enumeration

• Space is described as a regular array of cells (usually cubes). Each cell is called a **Voxel**
• A 3D object is represented as a list of filled voxels
Spatial Occupancy Enumeration

• **Pros:**
  – Easy to verify if a point (a voxel) is inside or outside an object
  – Boolean operations are easy to apply

• **Cons:**
  – Memory costs are high
  – Resolution is limited to size and shape of voxel
Quadtrees

- A **Quadtree** is a data structure enabling efficient storage of 2D data
- Completely full or empty regions are represented by one cell; recursive subdivision is used on the others
Octrees

• An Octree is a 3D generalization of a Quadtree
• Each node in an Octree has eight children rather than four
• Describes a recursive partitioning of a volume into cells that are completely full or empty
Binary Operations on Quad/Octrees

Notation:

- **P**: Internal Node (Partially full)
- **F**: Full Leaf Node
- **E**: Empty Leaf Node

Union:
- $P \cup P \rightarrow \text{Recursion on descendents}$
- $P \cup F = F$
- $P \cup E = P$

Intersection:
- $P \cap P \rightarrow \text{Recursion on descendents}$
- $P \cap F = P$
- $P \cap E = E$

Complement:
- $P^C \rightarrow \text{Recursion on descendents}$
- $F^C = E$
- $E^C = F$

Difference:
- $P - P \rightarrow \text{Recursion on descendents}$
- $P - F = E$
- $P - E = P$
Binary Operations on Quad/Octrees

A ∪ B

A ∩ B
Binary Space Partition Trees - BSP

- Each internal node represents a plane in 3D space.
- Each node has 2 children pointers one for each side of the plane.
- A leaf node represents a homogeneous portion of space - either “in” or “out”.
- Easy to determine if a point is inside or outside an object (recurse down the BSP tree).
Constructive Solid Geometry

• Combine simple primitives using Boolean operations and represent as a binary tree.
• To generate the object the tree is processed in a depth-first pass.
• Cons: representation is not unique
Constructive Solid Geometry

An object defined by a binary CSG tree
Boundary Representations

• A closed 2D surface defines a 3D object
• At each point on the boundary there is an “in” and an “out” side
• Boundary representations can be defined in two ways:
  – Primitive based. A collection of primitives forming the boundary (polygons, for example)
  – Freeform based (splines, parametric surfaces, implicit forms)
Boundary Representations

• A polyhedron is a solid bounded by a set of polygons
• A polyhedron is constructed from:
  – Vertices $V$
  – Edges $E$
  – Faces $F$
• Each edge must connect two vertices and be shared by exactly two faces
• At least three edges must meet at each vertex
Boundary Representations

• A **simple polyhedron** is one that can be deformed into a sphere (contains no holes)

• A simple polyhedron must satisfies *Euler's formula*:

\[ V - E + F = 2 \]

\[ \begin{align*}
V &= 8 \\
E &= 12 \\
F &= 6 \\
\end{align*} \]

\[ \begin{align*}
V &= 5 \\
E &= 8 \\
F &= 5 \\
\end{align*} \]

\[ \begin{align*}
V &= 6 \\
E &= 12 \\
F &= 8 \\
\end{align*} \]
Boundary Representations

• Euler’s formula can be generalized to a polyhedron with holes and multiple components

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

Where:
- \(H\) is the number of holes in the faces
- \(C\) is the number of separate components
- \(G\) is the number of pass-through holes (genus if \(C = 1\))
- \(V, E\) and \(F\) are respectively vertices, edges and faces

\[
24 - 36 + 15 - 3 = 2 \times (1 - 1)
\]