Outline: Surface Modelling

- Polygonal Meshes
- Solid modelling
- Primitive instancing
- Sweep representations
- Boundary representations
- Spatial partitioning representations
- Constructive solid geometry

Polygonal Meshes

- A polygonal mesh is a set of connected planar polygons which are used to represent the surface of an object.
- Work best for man made objects, but can use large number of polygons to approximate curves.
- Within a polygonal mesh each polygon edge is shared by only two polygons.
- An edge always connects two vertices.
- The polygon is a closed sequence of edges (which must belong to a polygon).

Polygonal Meshes

- Several possible representations with space-time tradeoff for each representation.
- The explicit representation defines each polygon P as:

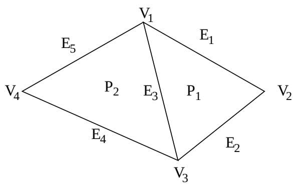
$$P = \{(x_1, y_1, z_1), \ldots, (x_n, y_n, z_n)\} \ ,$$

- The pointers to vertex list method uses a vertex list:

$$V = \{(x_1, y_1, z_1), \ldots, (x_n, y_n, z_n)\} \ ,$$

- Each polygon is represented as a series of pointers to vertices in the vertex list, $P = \{V_1^*, \ldots, V_n^*\}$ but unless special care is taken edges are drawn twice.

Polygonal Meshes



- Using pointers to edges list, we keep the vertex list V, but add an edge list E. Has structure: $E_i = \{V_j^*, V_k^*, P_r, P_l\}$, $P_m = \{E_s^*, E_t^*, E_u^* \ldots\}$. One of P_r or P_l can be the null or external polygon.
- For display the edges are drawn with transformations applied to the vertex list.

Polygonal Meshes - OpenGL

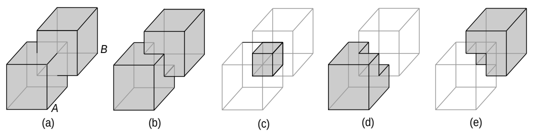
- The most simple way to implement polygonal meshes in OpenGL is to approximate the pointers to vertices structure. Define the vertices of your object and then use these to define the different polygons using `glBegin(.)` and `glEnd(.)`.
- OpenGL does the drawing, so no need to worry about edge lists.
- The tricky part is specifying the coordinates of the vertices and making sure they are given in the correct (anti-clockwise) order as viewed from the front of the polygon.
- In general use combinations of simple shapes and use transformations to get these to join up and make the object.
- Can use code I have supplied to compute the surface normals.

Solid modelling

- Important in CAD/CAM applications.
- What capabilities should a solid model have?
  - domain of representation.
  - unambiguous or complete.
  - unique and accurate.
  - closed under affine transformations.
  - compact and efficient.

Solid modelling

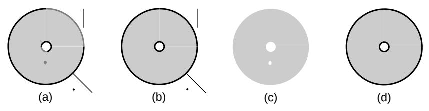
- We can apply Boolean operations:



- These allow us to represent complex shapes from combinations of simple objects.

Solid modelling

- When we use Boolean operations we need to take some care:



  - (a) We need to regularise the result.
  - (b) Closure: union of the set and it's boundary.
  - (c) The interior set.
  - (d) Regularisation: closure of the set's interior points.

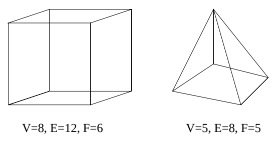
Solid modelling

- Some common methods used in solid modelling are:
  - Primitive instancing
  - Sweep representations (translational sweeps or rotational sweep)
  - Boundary representations
  - Spatial partitioning representations
  - Constructive solid geometry
- We will not cover any of these in great depth.

Boundary representations

- Similar to polygonal meshes but used for solids - often called b-reps.
- Most can only represent shapes that have boundaries that are 2-manifolds.
- A polyhedron (which belongs to the set of objects having 2-manifolds) is a solid which is bounded by a set of polygons whose edges belong to only one other polygon.
- A simple polyhedron has no holes in it - that is it can be transformed into a sphere without breaking any of the connectivity.
- Can be used with curved surfaces.

B-reps

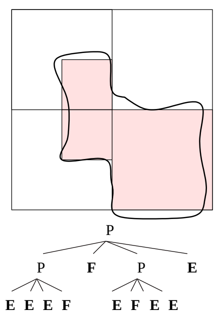


- Euler's formula tells us that for a simple polyhedron with V vertices, E edges and F faces, $V - E + F = 2$.
- This gives us necessary conditions for an object to be a simple polygon, but not sufficient conditions.
- Sufficient conditions are given by ensuring that each edge only connects two vertices and is shared by only two faces, faces do not inter-penetrate and at least three faces meet at each vertex.

Spatial partitioning representations

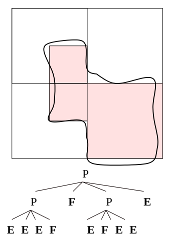
- Divide the solid to be represented into a number of primitive solid objects.
- Cell decomposition represents the object as a number of small, simple cells.
- Spatial occupancy enumeration: only one type of cell is used on a fixed, regular grid.
- The cells are often referred to as voxels (volume elements).
- Amount of information need grows as n^3 if n is the number of voxels along each axis however hierarchical methods can be used.

Octrees



- Example shows the 2D equivalent, the quadtree.

Octrees

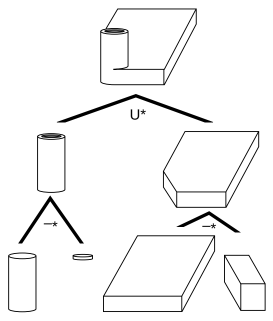


- Octrees apply a binary subdivision in a divide-and-conquer algorithm.
- The number of nodes in an octree is proportional to the objects surface area, which is of order n^2 rather than n^3.

Binary Space-Partition Trees

- Binary Space-Partition trees divide space into two using arbitrary planes.
- Each internal node of the BSP tree is associated with a plane and has two children, inside and outside (for each side of the plane).
- Used in visible surface determination.

Constructive solid geometry



- Constructive solid geometry the solid is defined using the regularised Boolean operations on some simple primitives.

Which model to use?

- Depends upon:
  - application,
  - types of objects,
  - cost of storage / processing.
- Also need to initialise the objects - this can be very complex.

Summary

- Having finished this lecture you should:
  - be aware of the different methods used to represent solid objects;
  - be able to implement polygon meshes;
  - be able to design solid models in OpenGL.
- This is not vital to the course but is included for completeness.