Building models

Charalambos Poullis

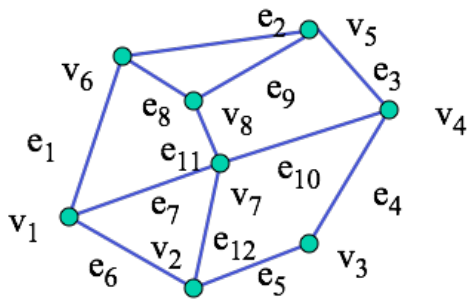
Department of Computer Science & Software Engineering
Faculty of Engineering & Computer Science

September 18, 2018

Lecture Overview

Mesh

A mesh is a set of polygons that share vertices and edges which describe the shape of a geometric object. Meshes can be simple consisting only of a few primitives, or very complex. Consider the following mesh,



- there are 8 nodes and 12 edges
- 5 interior polygons
- 6 interior (shared) edges
- each vertex has a location $v_i = (x_i y_i z_i)$

Simple representation

Define each polygon by the geometric locations of its vertices:

glm::vec3 v1 = glm::vec3(x1, y1, z1);
glm::vec3 v2 = glm::vec3(x2, y2, z2);
...

$\rightarrow$ inefficient and unstructured

Inward and outward facing polygons

The order v_1, v_2, v_7 and v_2, v_7, v_1 are equivalent in that the same polygon will be rendered by OpenGL but the order v_1, v_7, v_2 is different

The first two describe outwardly facing polygons

Use the right-hand rule = counter-clockwise encirclement of outward-pointing normal

OpenGL can treat inward and outward facing polygons differently

Geometry vs topology

Geometry: locations of the vertices

Topology: organization of the vertices and edges

Example: a polygon is an ordered list of vertices with an edge connecting successive pairs of vertices and the last to the first

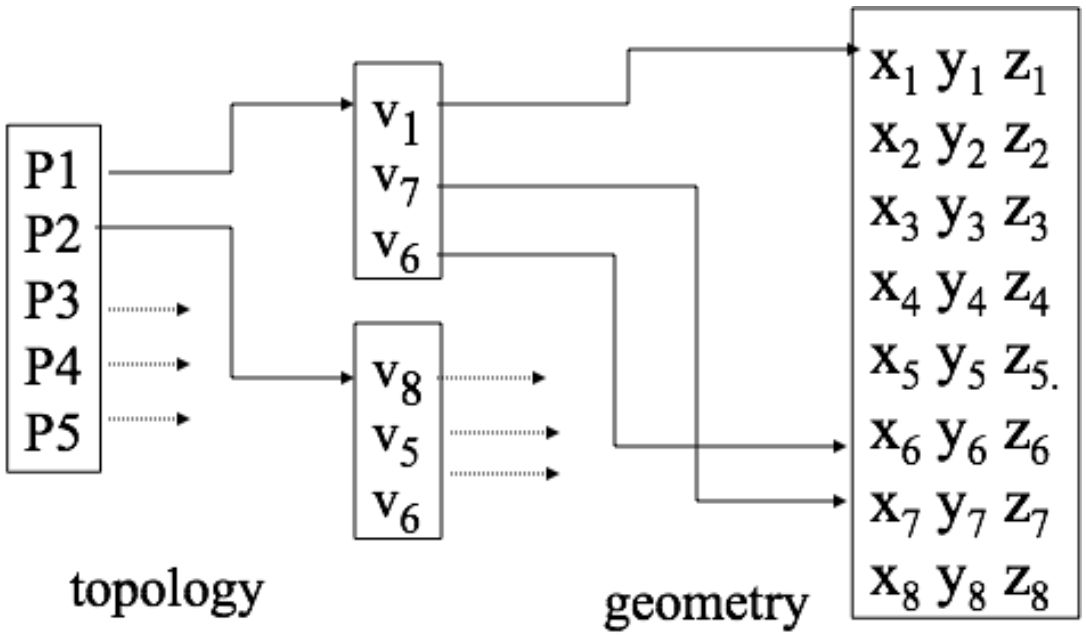
Topology holds even if geometry changes

Vertex lists

Put the geometry in an array

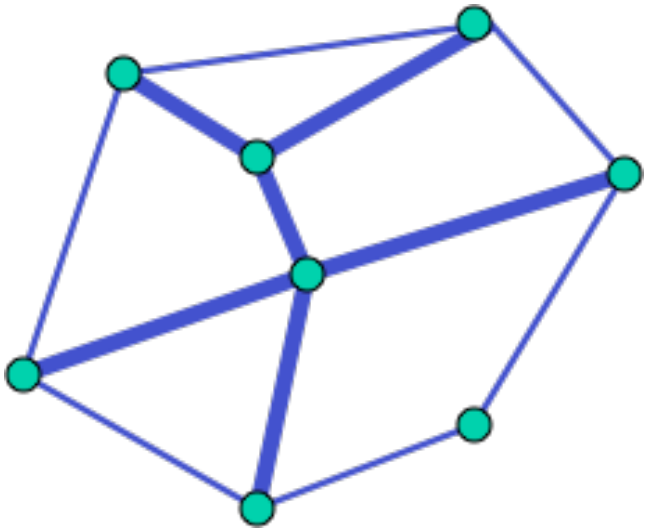
Use pointers from the vertices into this array

Introduce a polygon list

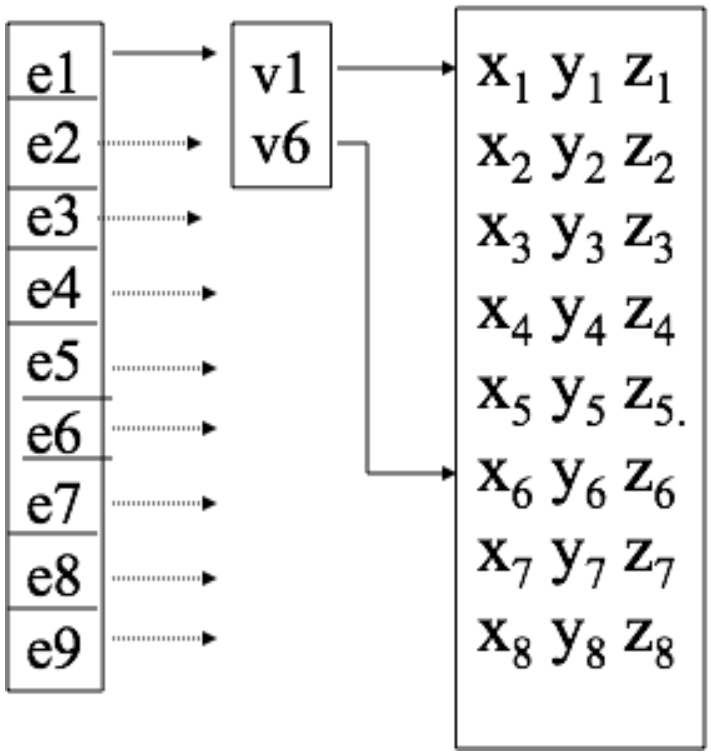


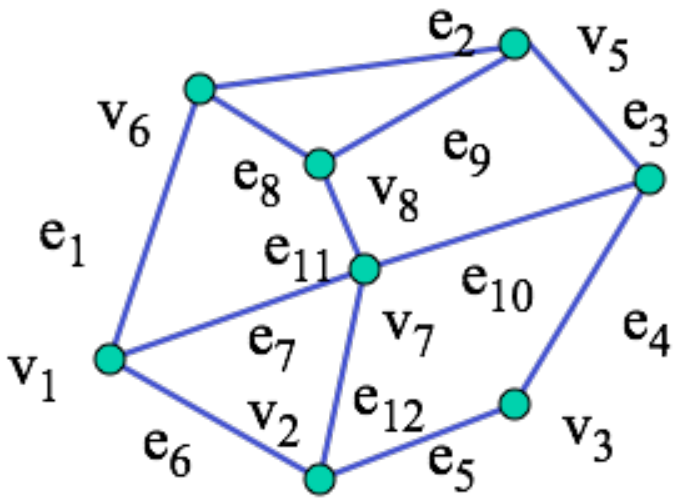
Shared edges

Vertex lists will draw filled polygons correctly but if we draw the polygon by its edges, shared edges are drawn twice



Edge lists





Note polygons are not represented

Vertex array objects (VAO)

OpenGL provides a facility called vertex arrays that allows us to store array data in the implementation attributes e.g. position, color, texture coordinates, indices, normals

Live demo

- VAO, VBO, EBO