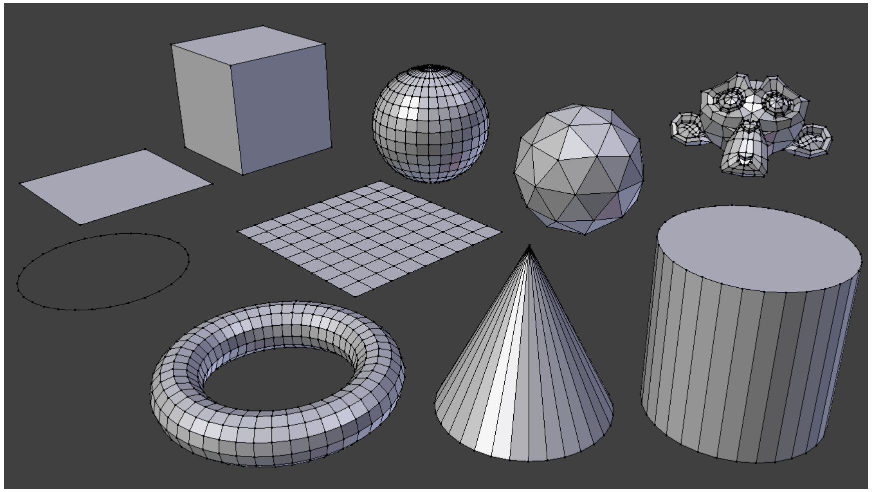
CMSC427
Parametric surfaces and polygonal meshes

Note

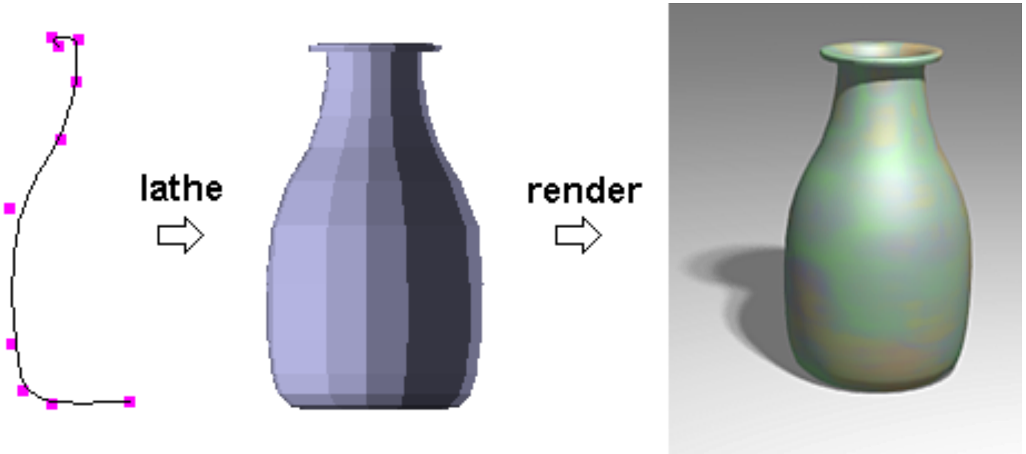
- These slides are incomplete
- See accompanying PDF with detailed outline
- Will develop many equations in class
- Reading later to supplement

Moving to 3D

- Polygonal meshes
  - Set of standard shapes in Blender

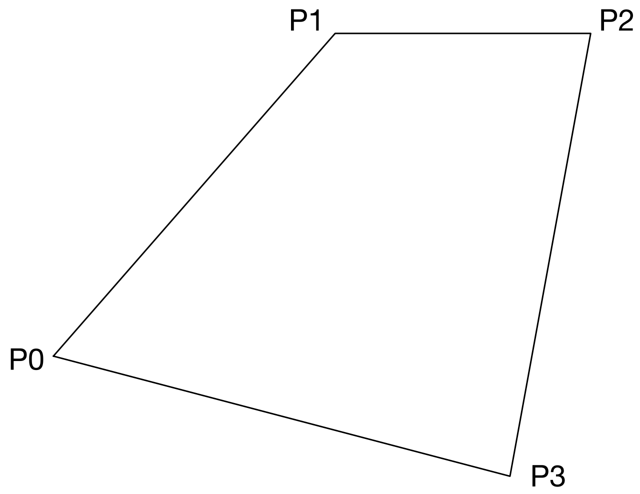


- And how to create them
  - And store them
  - And draw them



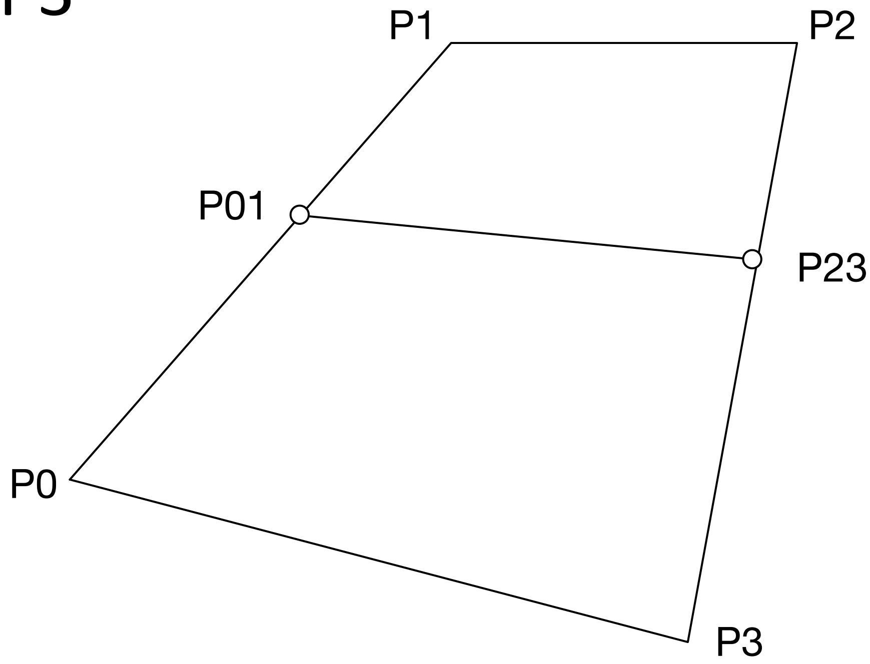
Bilinear patch

- Blending of four 3D points
- Ruled surface
  - Swept out by sequence of lines



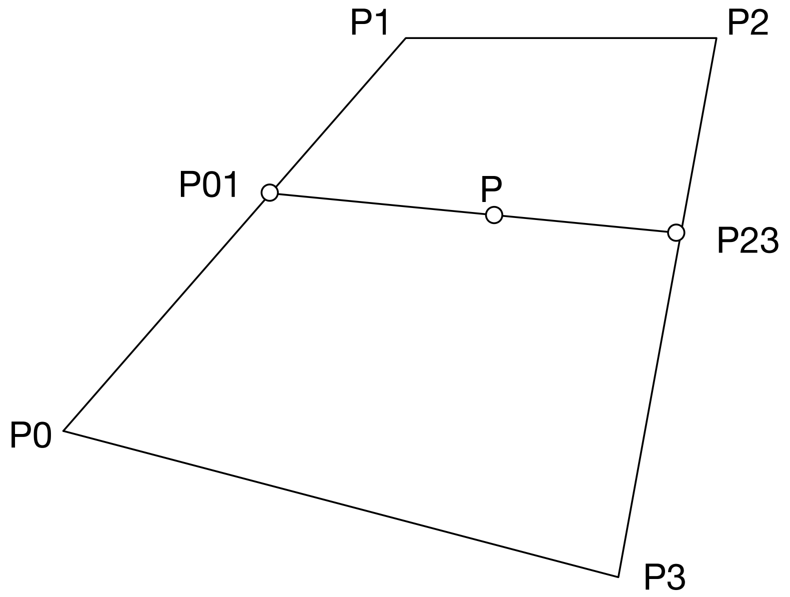
Bilinear patch

- Blend simultaneously along two lines
- P01 = t(P1-P0) + P0
- P23 = t(P2-P3) + P3
- Same t in [0,1]



Bilinear patch

- Blend simultaneously along two lines
- P01 = tP1 + (1-t)P0
- P23 = tP3 + (1-t)P2
- Same t in [0,1]

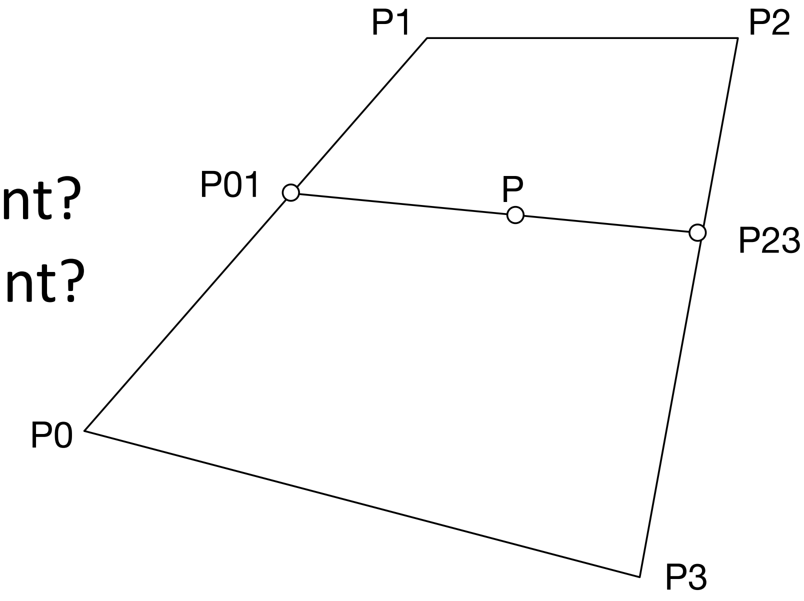


- Then blend between the two lines

- P = sP23 + (1-s)P01
- P = s(tP1 + (1-t)P0) + (1-s)(tP3 + (1-t)P2)

Bilinear patch

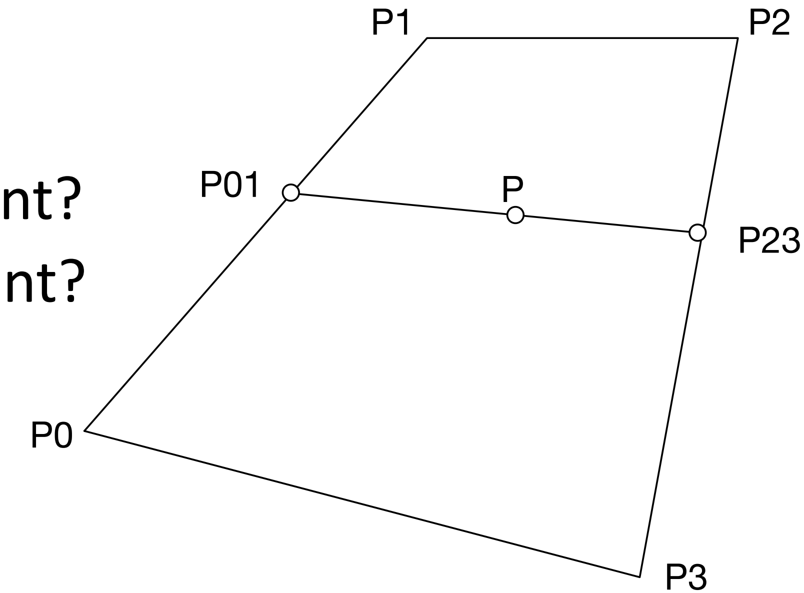
- Questions
  - What order polynomial?
  - Convex combination?
  - What is drawn if t is constant?
  - What is drawn if s is constant?



- $P = s(tP1 + (1\text{-}t)P0) + (1\text{-}s)(tP3 + (1\text{-}t)P2)$

Bilinear patch

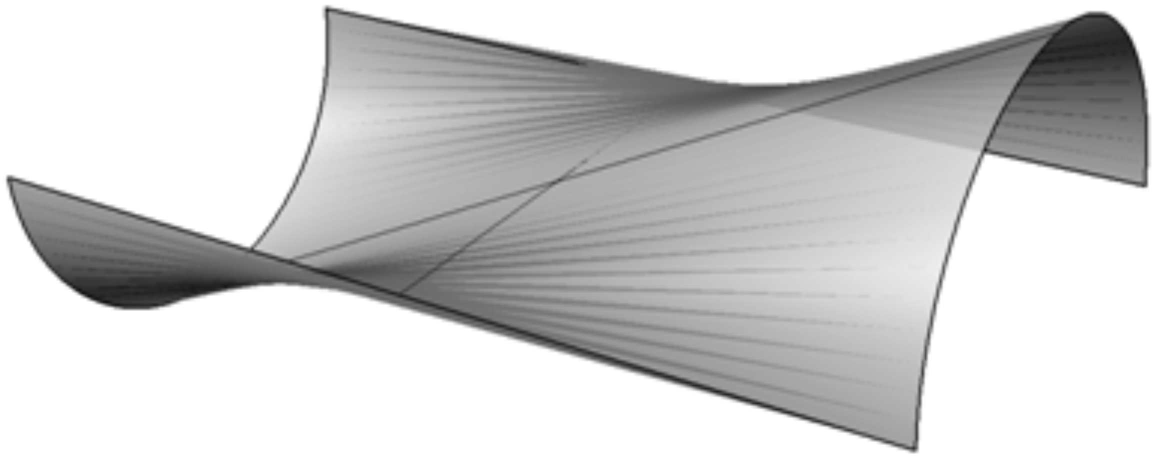
- Questions
  - What order polynomial?
  - Convex combination?
  - What is drawn if t is constant?
  - What is drawn if s is constant?



- $P = s(tP1 + (1-t)P0) + (1-s)(tP3 + (1-t)P2)$
- $P = stP1 + s(1-t)P0 + (1-s)tP3 + (1-s)(1-t)P2$

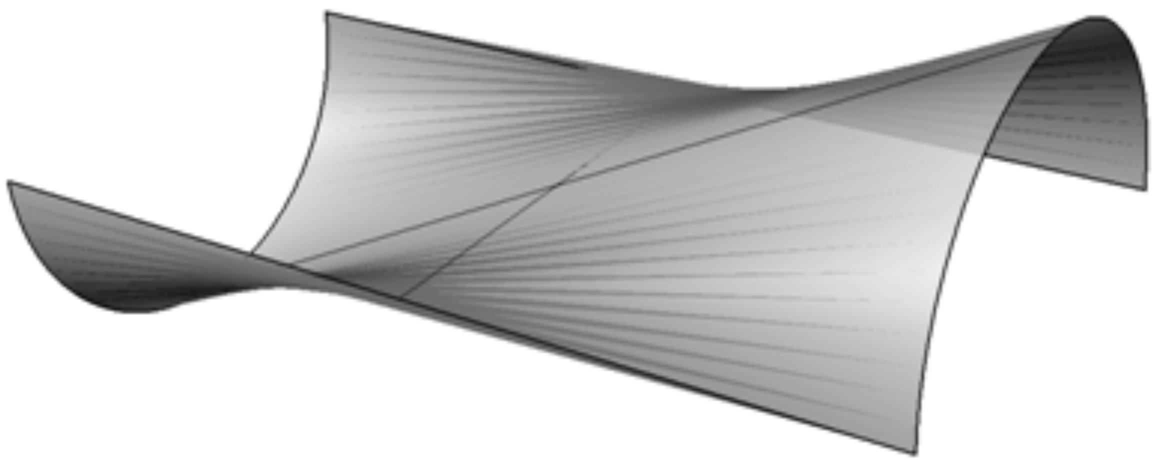
Coons patch

- What's happening in this surface?



Coons patch

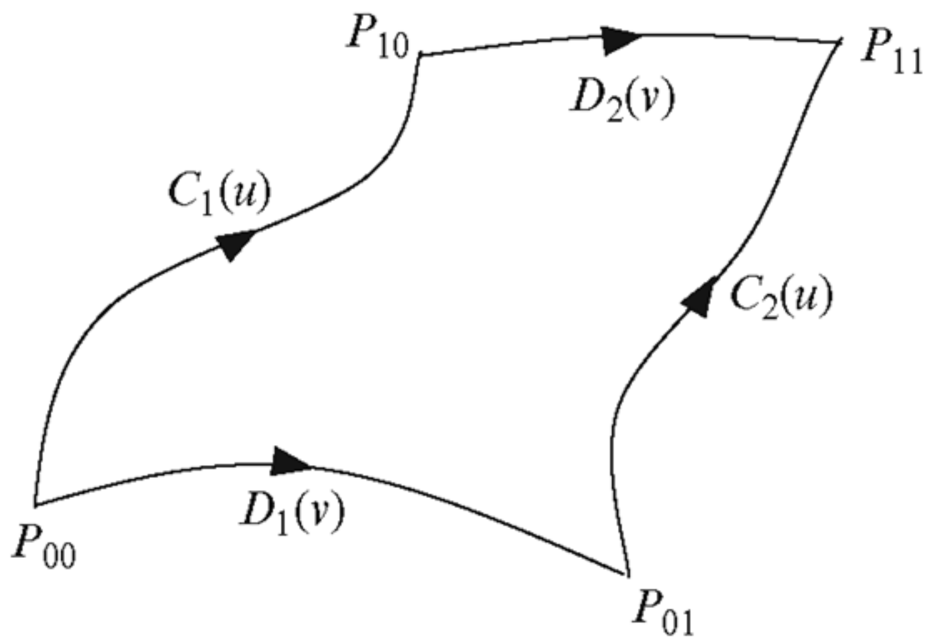
- What's happening in this surface?



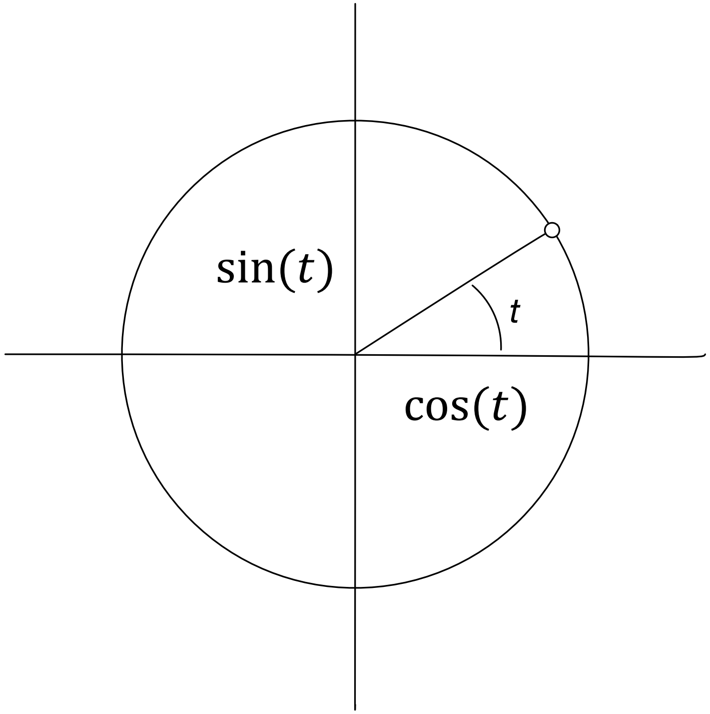
- Blending two arcs
  - Is this a ruled surface?

Coons patch

- Blend four arbitrary curves
- Here C1, C2, D1, D2



Circle with trig: review



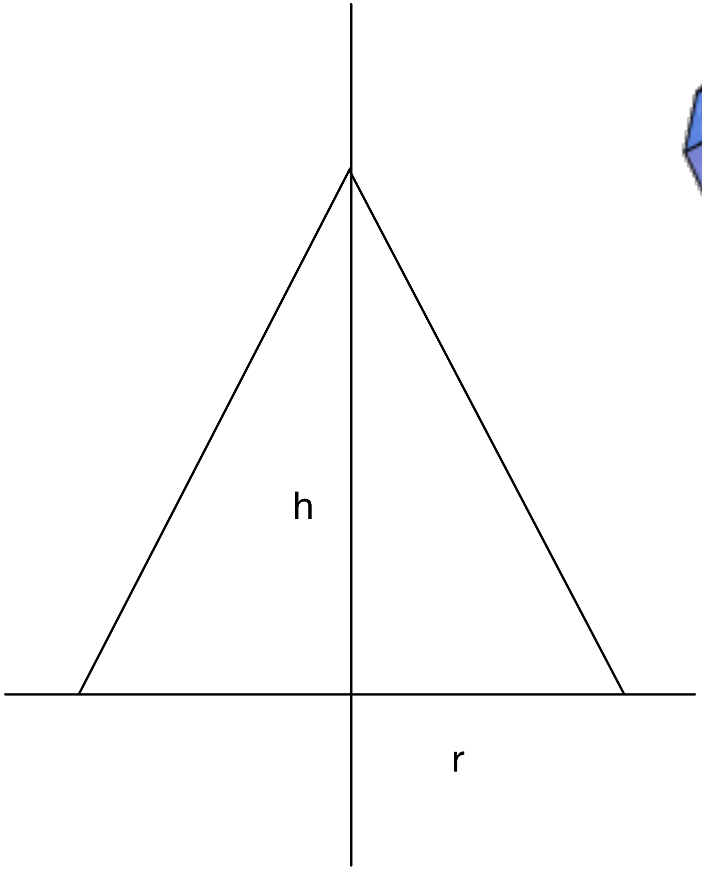
Parametric equation

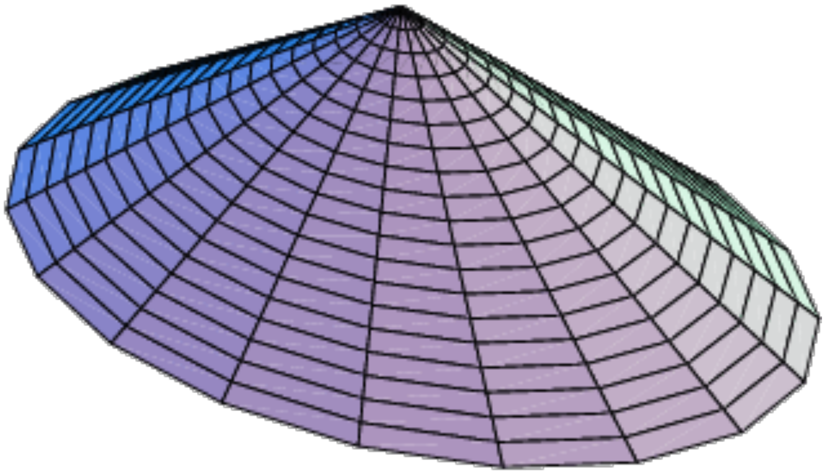
$$x = R\cos(t)$$

$$y = R\sin(t)$$

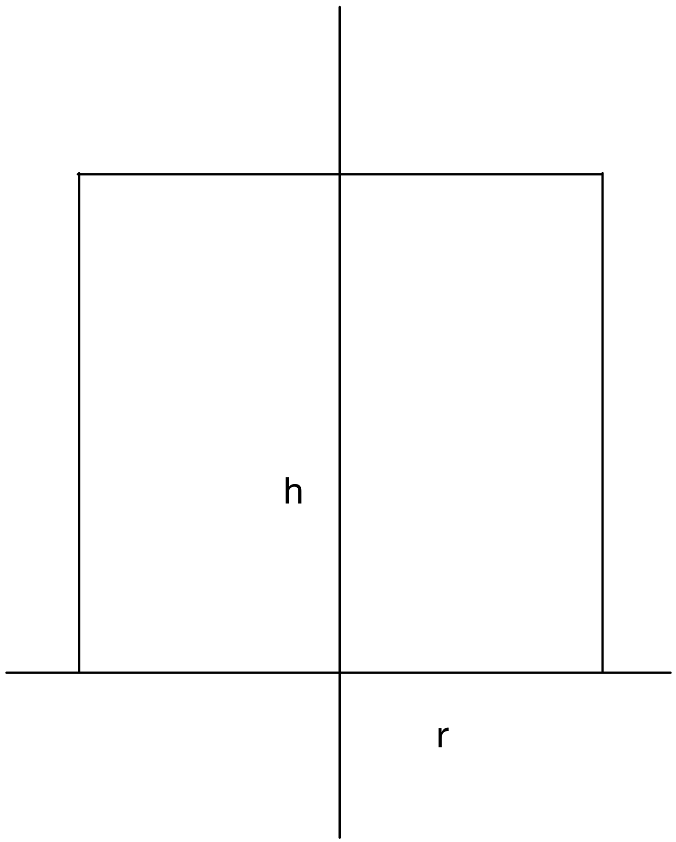
$$0 \leq t \leq ??$$

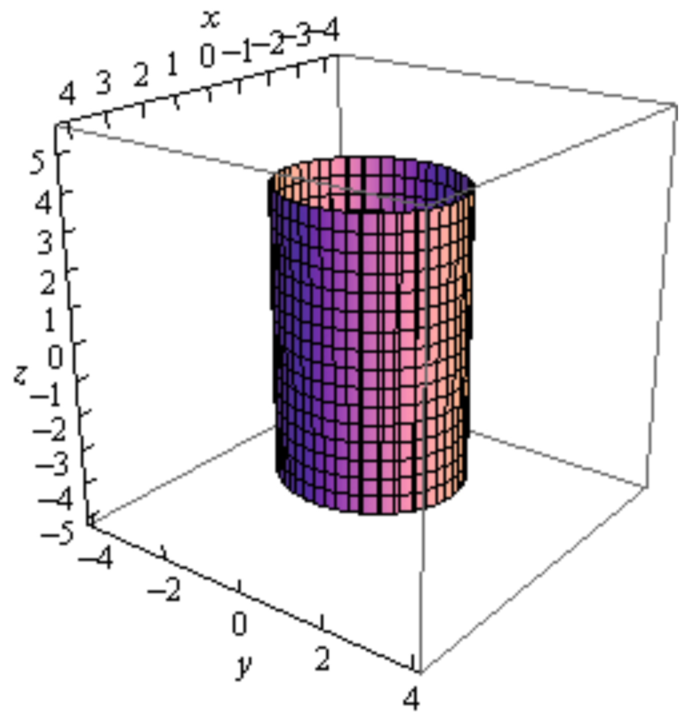
Parametric cone





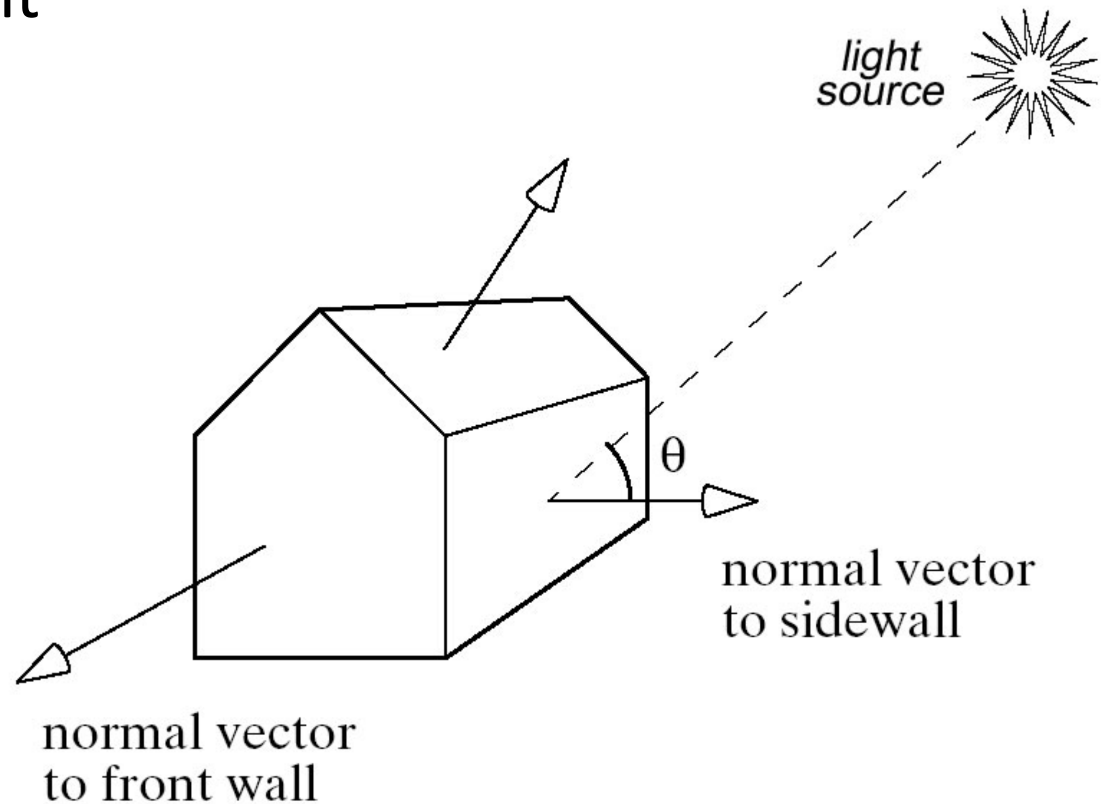
Parametric cylinder





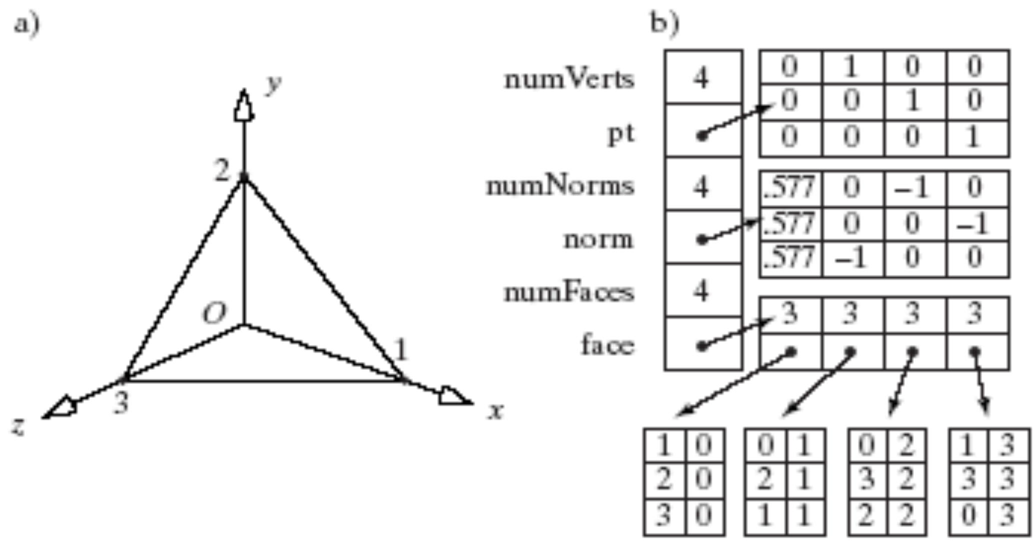
Rendering faces: need location and normal

- Need distance and orientation relative to lights to compute reflected light



Polygonal mesh

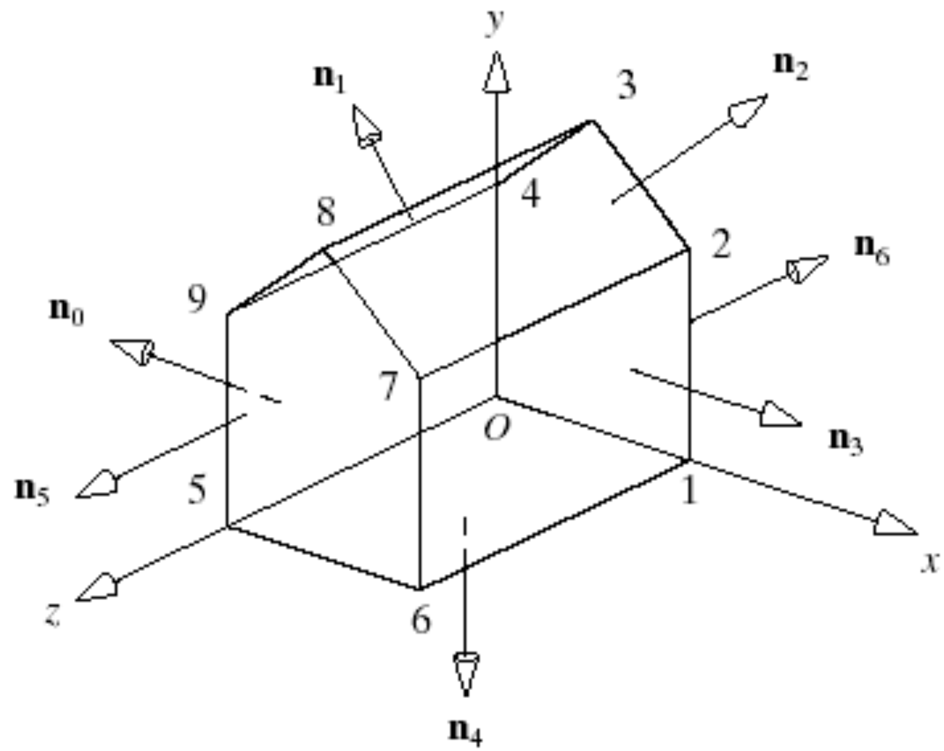
- Simplest mesh: tetrahedron
- Indexed mesh representation
  - Vertex list
  - Normal list
  - Face list



- Non-indexed representation
  - List of faces with repeated vertices

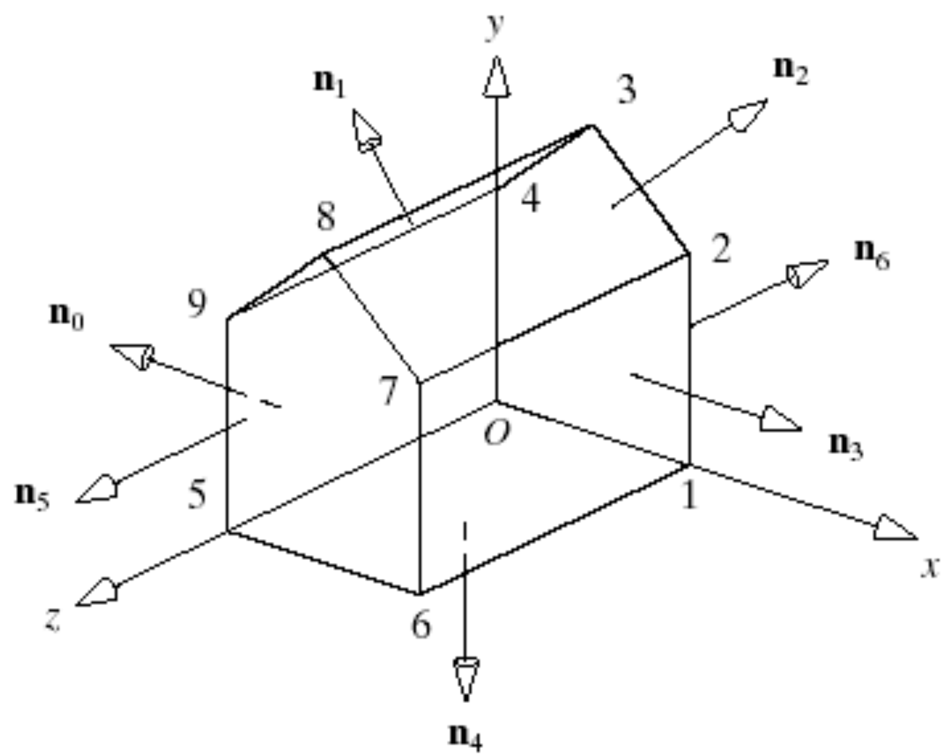
Polygonal mesh

- Hill's barn
- 10 vertices
- 7 faces
- 7 normals



Polygonal mesh

- Hill's barn
- 10 vertices
- 7 faces
- 7 normals

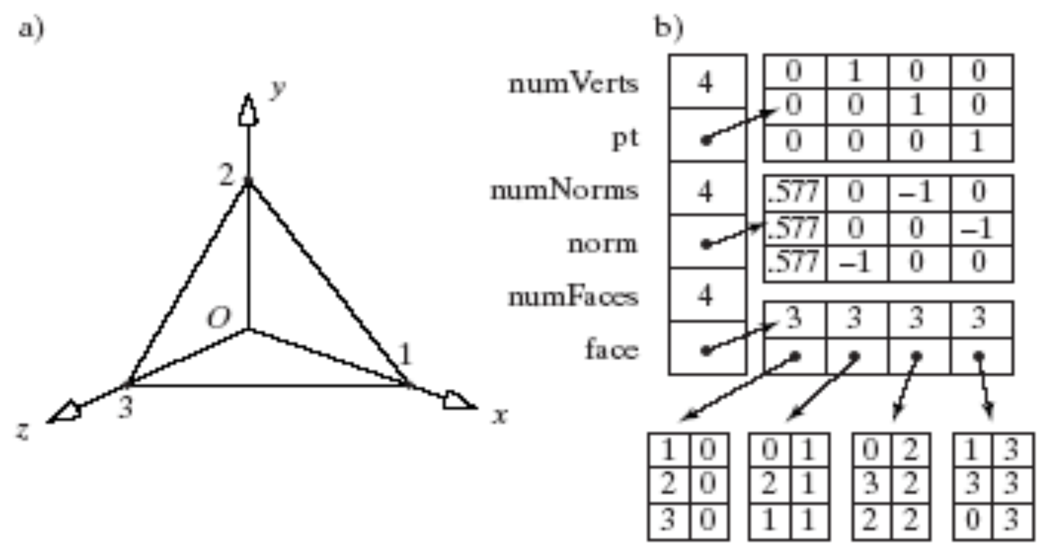


- Solution for one face:
  - Face vertices (CCW):
    - 5 6 7 8 9
  - Face normals:
    - 5 5 5 5 5

- Alternative: triangulate
  - Face1: 5 6 7, Face2: 5 7 9, Face 3: 7 8 9

Drawing mesh

- Draw as points: iterate through points
- Draw as lines: iterate through adj. pts. in faces
  - Problem?
  - Alternative: add edge list to structure
  - Alternative: better link faces to avoid redundancy
- Draw as ”solid”: iterate through faces



File formats

- STL
  - https://en.wikipedia.org/wiki/STL_(file_format)
- OBJ
  - https://en.wikipedia.org/wiki/Wavefront_.obj_file
- Many others
- Not hard to generate your own STL files

Meshlab

- Free viewing software: Meshlab
  - (http://www.meshlab.net)
  - Good for viewing, repairing, decimating meshes

- Sources of 3D mesh models:
  - SketchFab (https://sketchfab.com)
  - Thingiverse (https://www.thingiverse.com)
  - Stanford repository (http://graphics.stanford.edu/data/3Dscanrep/)

- Std Examples: Utah teapot, Stanford bunny

Generating polygonal mesh from parametric surface

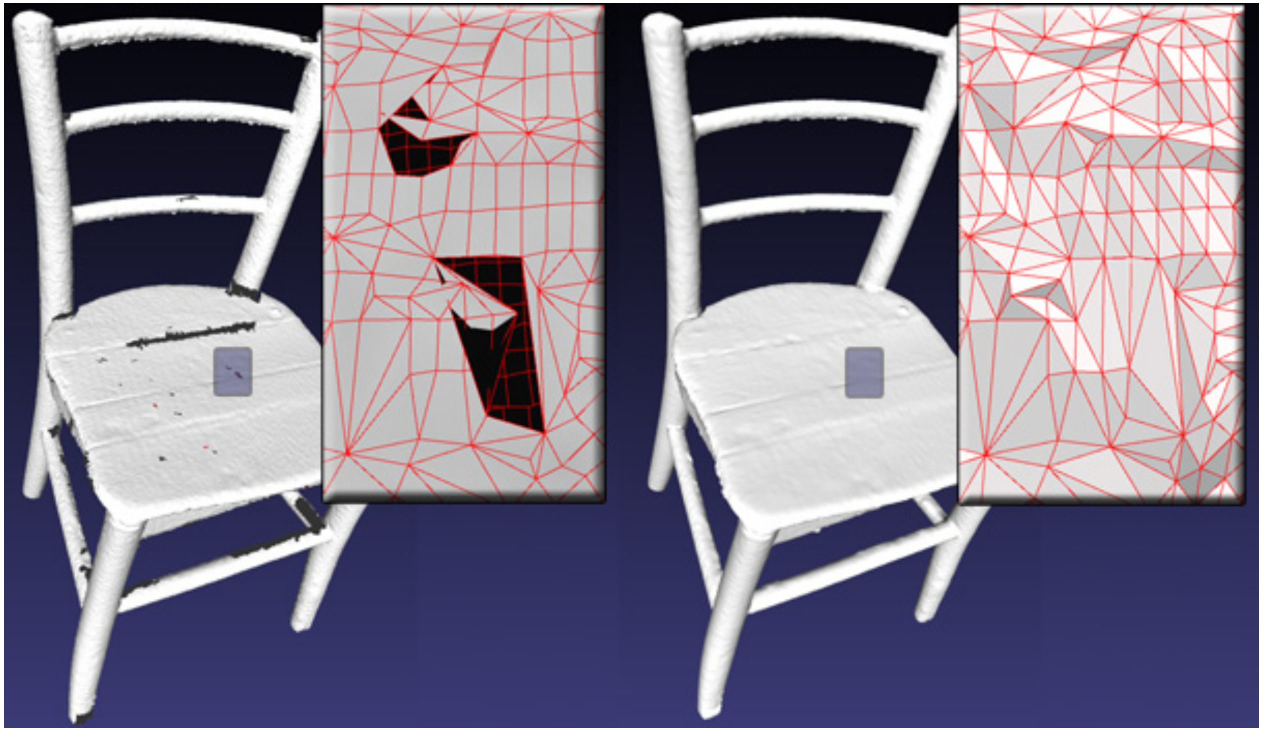
- Step 1:

- Sources of 3D mesh models:
  - SketchFab (https://sketchfab.com)
  - Thingiverse (https://www.thingiverse.com)
  - Stanford repository (http://graphics.stanford.edu/data/3Dscanrep/)

- Std Examples: Utah teapot, Stanford bunny

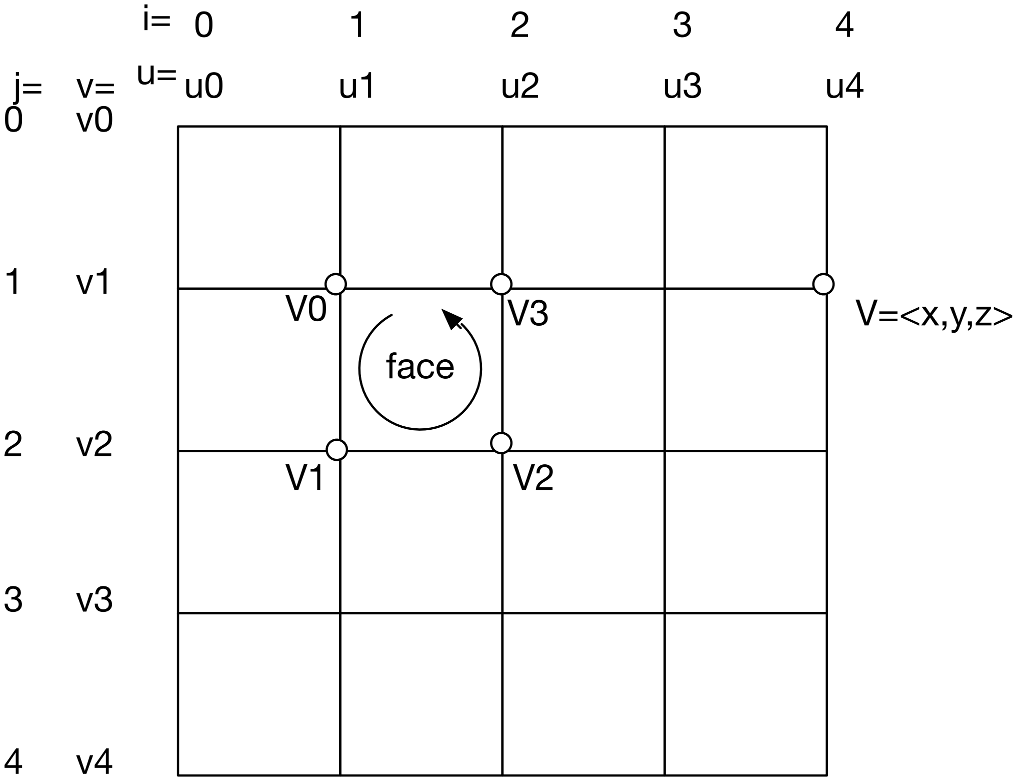
Topological properties of meshes

- Is a mesh well connected?
- Any flaws?
- More later



Generating mesh from parametric surface

- Given parametric surface
  - $P(u,v) = < x(u,v), y(u,v), z(u,v) >$
- Generate mesh
- Steps:
  - 1. Set # divisions in u,v as n, m
  - 2. Generate u,v in for loops
  - 3. Store in 2D array of 3D points
  - 4. From array generate faces



Computing normal vectors for mesh

- Approach 1: Cross product of numeric data
  - Find v1 and v2 from vertices (which?)
  - N = v1 x v2
  - Less arbitrary: Newell's method

- Approach 2: Partial derivatives of parametric curve
  - Given vector P(u,v) = < x(u,v), y(u,v), z(u,v)>
  - Derive vectors dU = dP(u,v)/du and dV = dP(u,v)/dv
  - N = dU x dV

- Approach 3: Gradient vector of implicit surface
  - Given implicit function f(x,y,z)
  - Derive gradient < df/dx, df/dy, df/dz >

Creating polygonal meshes: summary

- Fixed shapes.
  - Any shape based on idiosyncratic data, such as the exact shape of a stone, foot, sculpture, etc. All hard-coded, some from real world data collection
- Regular polyhedron
  - Cubes, tetrahedrons, icosahedrons, dodecahedrons, ...
- Operations that create shapes
  - Extrusion
  - Lathing (surfaces of rotation)
  - Surface subdivision
- Parametric shapes (related to operations)
  - Bilinear patches, quadrics, superellipses, etc.