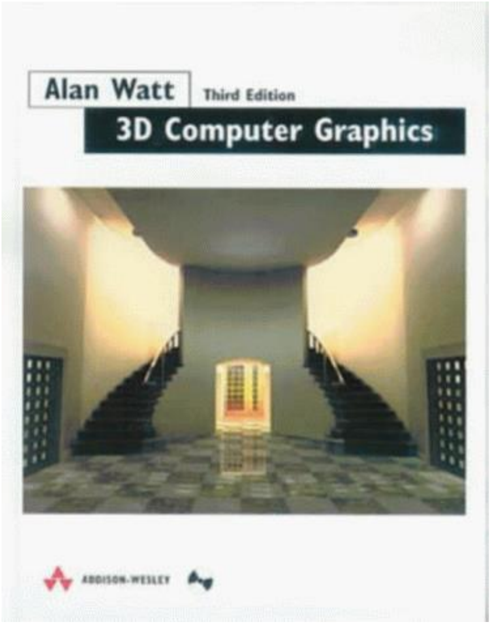
COM3503/4503/6503: 3D Computer Graphics

Lecture 4: Polygon meshes



Dr. Steve Maddock
Room G011, Regent Court
s.maddock@sheffield.ac.uk

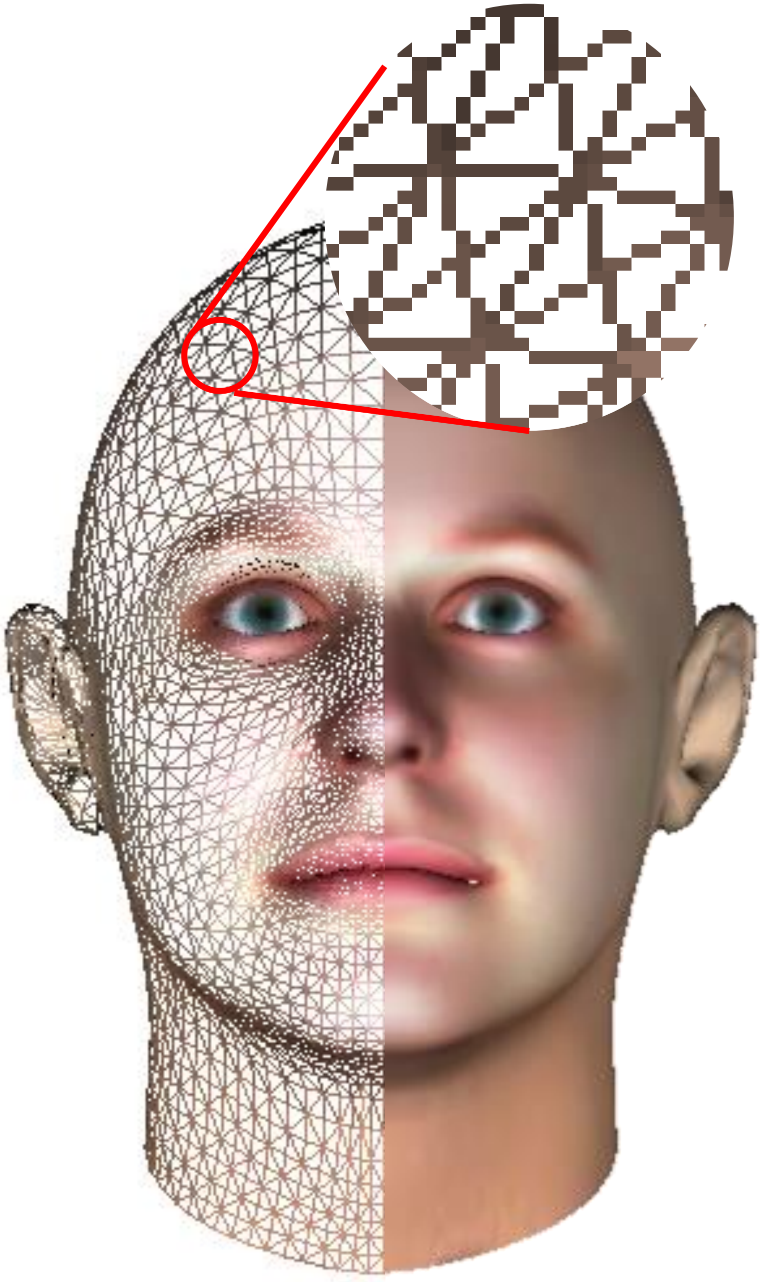
1. Representation of 3D objects

- Many alternative representations
  - Polygons, parametric patches, CSG, space subdivision, implicit representation, etc.
- The representation will determine:
  - Data structure and form of processing algorithms;
  - Ease of editing;
  - Cost of processing an object;
  - Final appearance of an object.

Kajiya, J.T. (1992). foreword, p.xi in 'Snyder, J.M. (1992). Generative Modeling for Computer Graphics and CAD. Academic Press'.

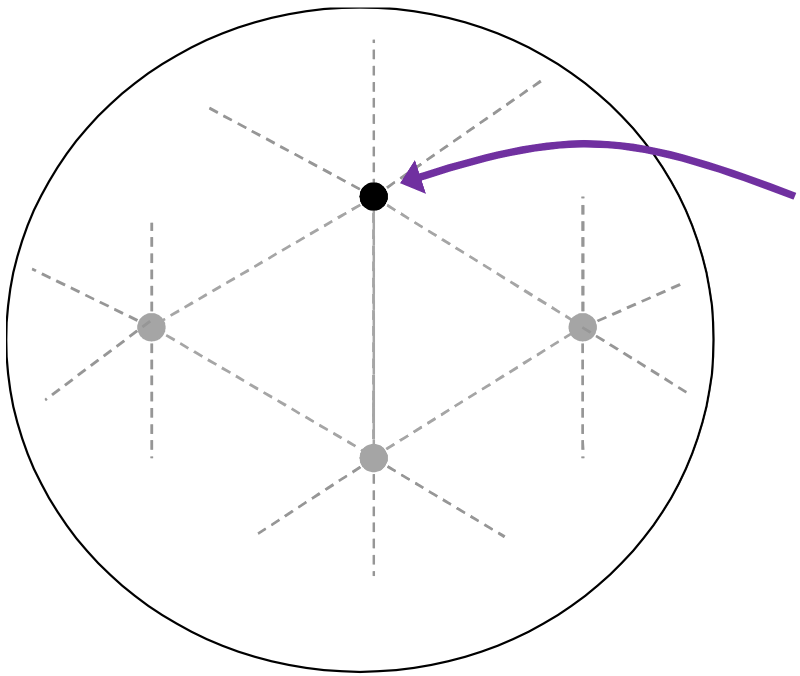
2. Polygons

- An object is represented by a mesh of polygonal facets
  - Set of points and connectivity information
- Most common representation in computer graphics is a mesh of triangles
- Referred to as a Boundary Representation or B-rep technique
- For curved surfaces, this is an approximation



3. Alternative polygon data structures

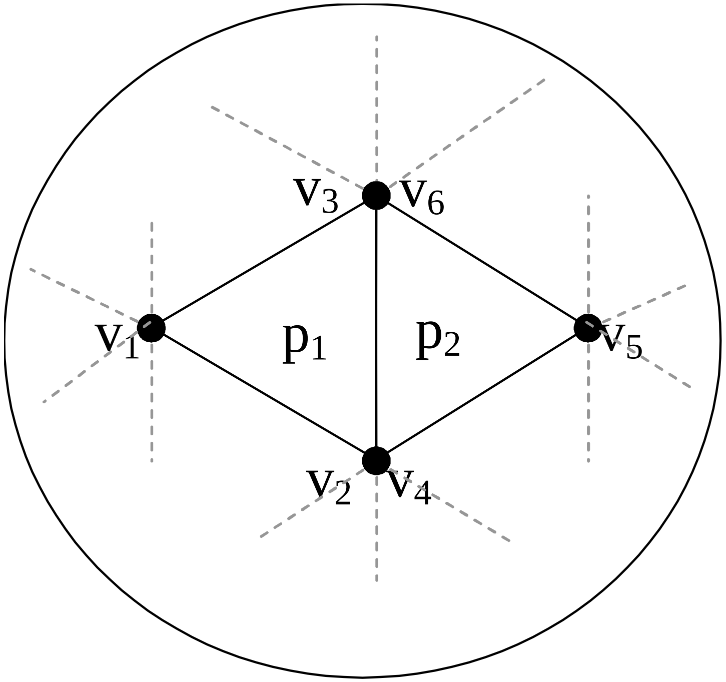
- What data structure should be used?



$v_i = (x_i, y_i, z_i)$
$i = 1..N$ (or $i=0..N-1$)
N = number of vertices

3.1 Polygons index the vertices

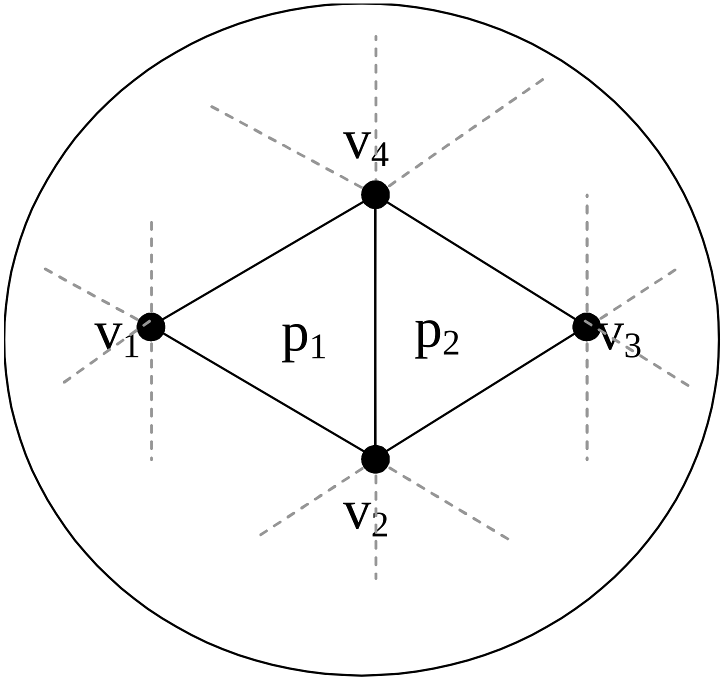
- Polygons are individual entities
- Vertices are duplicated
- No representation for shared edges or vertices



$$V = (v1,v2,v3,v4,v5,v6, \ldots)$$
$$= ((x_1,y_1,z_1), \ldots)$$
$$p1 = (v1,v2,v3)$$
$$p2 = (v4,v5,v6)$$

3.1.1 Vertices not duplicated

- Same vertex indexed by more than one polygon
- Inefficient to find polygons which share an edge



$$V = (v1,v2,v3,v4, \ldots)$$
$$= ((x_1,y_1,z_1), \ldots)$$
$$p1 = (v1,v2,v4)$$
$$p2 = (v2,v3,v4)$$

See Element Buffer Objects example in Week 1 lab class

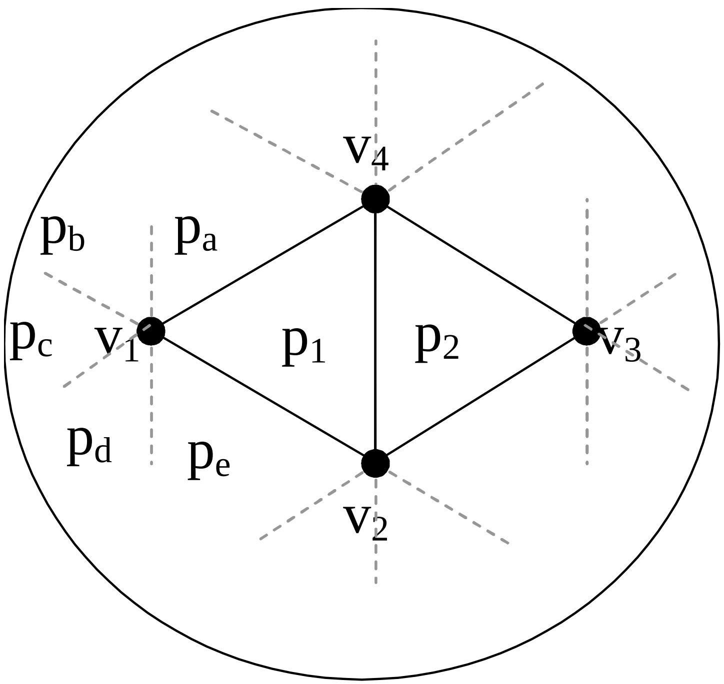
3.1.1 Vertices not duplicated

- *Example*: .obj file for a sphere represented as a mesh of triangles
- v 0.0 0.0 0.5
  - A vertex and its x,y,z, values
  - The first vertex is vertex number 1
- Optional:
  - vt – vertex texture coordinates
  - vn – vertex normal (x,y,z)
- f a/b/c a/b/c a/b/c
  - A polygon face with up to three index values for each vertex
  - vertex index / vertex texture index/ vertex normal index
  - Optional: vertex texture number and vertex normal number

```
# object Sphere01 to come
...
#
v 0.0 0.0 0.5
v 0.0 0.294 0.405
v -0.173 0.23799999 0.405
v -0.28 0.091 0.405
…
# 42 vertices
…
s 1
f 1 2 3
f 1 3 4
f 1 4 5
f 1 5 6
…
# 80 faces
…
```

3.2 Face-vertex meshes

- Most widely used data structure
- Polygons index vertices
- Vertices index polygons



v1 = (x_1,y_1,z_1,
p1,pa,pb,pc,pd,pe)
v2 = (…)
…
p1 = (v1,v2,v4)
p2 = (v2,v3,v4)
…

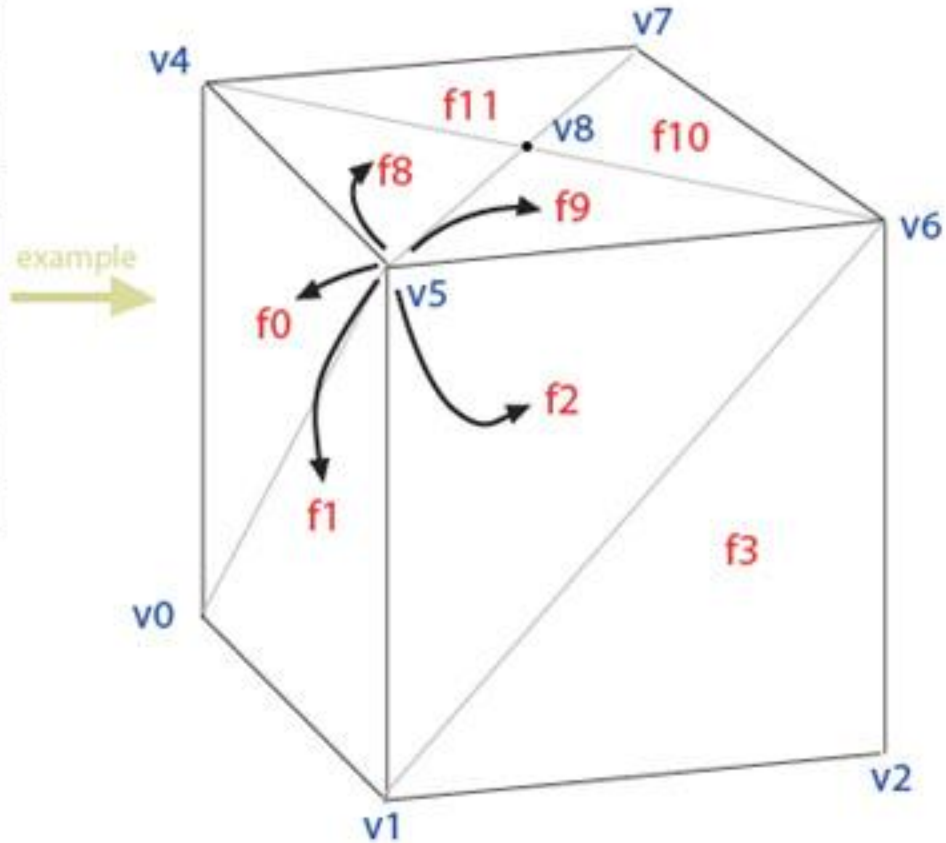
Face-Vertex Meshes

Face List

f0	v0 v4 v5
f1	v0 v5 v1
f2	v1 v5 v6
f3	v1 v6 v2
f4	v2 v6 v7
f5	v2 v7 v3
f6	v3 v7 v4
f7	v3 v4 v0
f8	v8 v5 v4
f9	v8 v6 v5
f10	v8 v7 v6
f11	v8 v4 v7
f12	v9 v5 v4
f13	v9 v6 v5
f14	v9 v7 v6
f15	v9 v4 v7

Vertex List

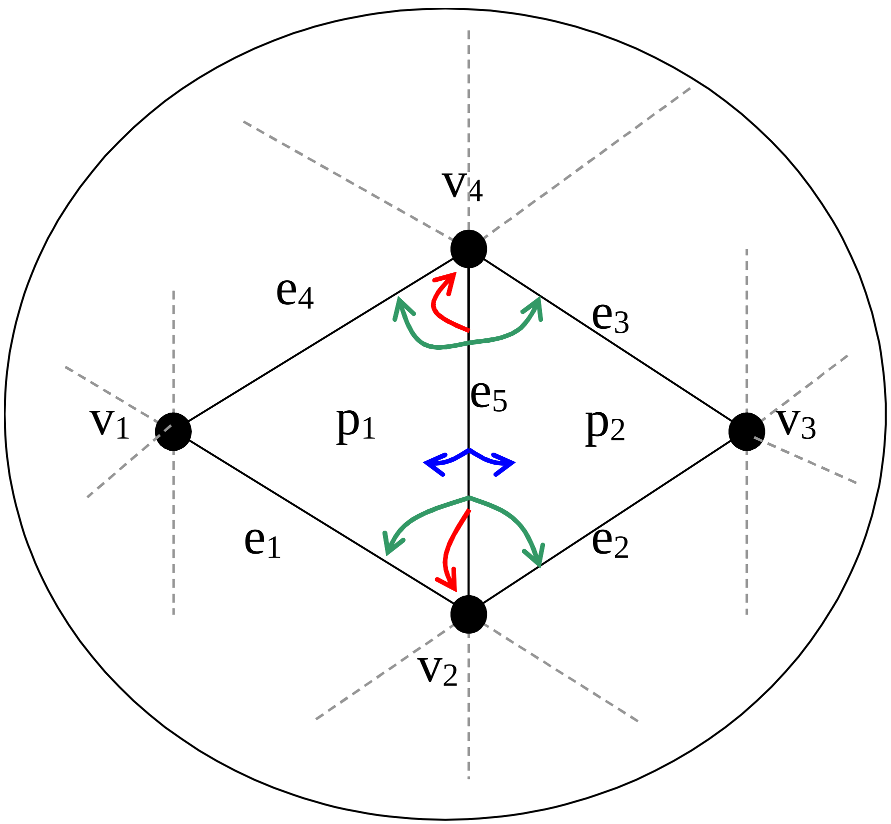
v0	0,0,0	f0 f1 f12 f15 f7
v1	1,0,0	f2 f3 f13 f12 f1
v2	1,1,0	f4 f5 f14 f13 f3
v3	0,1,0	f6 f7 f15 f14 f5
v4	0,0,1	f6 f7 f0 f8 f11
v5	1,0,1	f0 f1 f2 f9 f8
v6	1,1,1	f2 f3 f4 f10 f9
v7	0,1,1	f4 f5 f6 f11 f10
v8	.5,.5,0	f8 f9 f10 f11
v9	.5,.5,1	f12 13 14 15



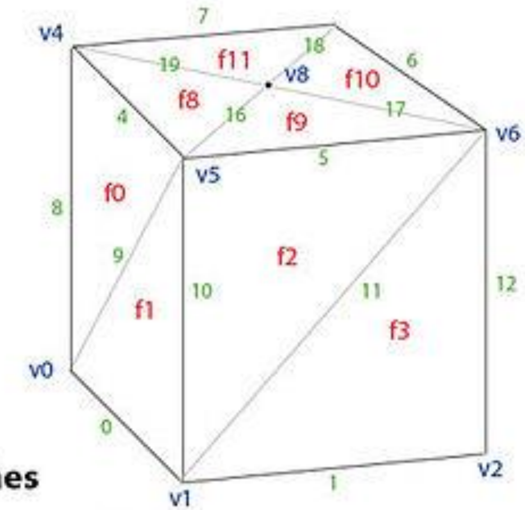
http://en.wikipedia.org/wiki/File:
Mesh_fv.jpg

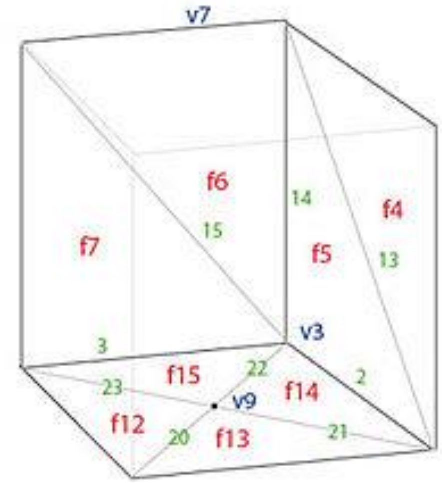
3.4 Winged edge meshes

- Widely used in modelling programs
- Flexible mesh geometry change
- Large storage requirements and increased complexity



An edge points to its two vertices, 4 of its adjoining edges (nearest CW and nearest CCW) and its adjacent polygons.
A vertex points to its connected edges.
A polygon points to its edges.





Winged-Edge Meshes

Face List

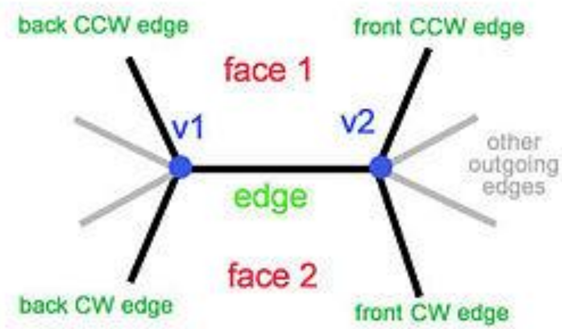
f0	4 8 9
f1	0 10 9
f2	5 10 11
f3	1 12 11
f4	6 12 13
f5	2 14 13
f6	7 14 15
f7	3 8 15
f8	4 16 19
f9	5 17 16
f10	6 18 17
f11	7 19 18
f12	0 23 20
f13	1 20 21
f14	2 21 22
f15	3 22 23

Edge List

e0	v0 v1	f1 f12	9 23 10 20
e1	v1 v2	f3 f13	11 20 12 21
e2	v2 v3	f5 f14	13 21 14 22
e3	v3 v0	f7 f15	15 22 8 23
e4	v4 v5	f0 f8	19 8 16 9
e5	v5 v6	f2 f9	16 10 17 11
e6	v6 v7	f4 f10	17 12 18 13
e7	v7 v4	f6 f11	18 14 19 15
e8	v0 v4	f7 f0	3 9 7 4
e9	v0 v5	f0 f1	8 0 4 10
e10	v1 v5	f1 f2	0 11 9 5
e11	v1 v6	f2 f3	10 1 5 12
e12	v2 v6	f3 f4	1 13 11 6
e13	v2 v7	f4 f5	12 2 6 14
e14	v3 v7	f5 f6	2 15 13 7
e15	v3 v4	f6 f7	14 3 7 15
e16	v5 v8	f8 f9	4 5 19 17
e17	v6 v8	f9 f10	5 6 16 18
e18	v7 v8	f10 f11	6 7 17 19
e19	v4 v8	f11 f8	7 4 18 16
e20	v1 v9	f12 f13	0 1 23 21
e21	v2 v9	f13 f14	1 2 20 22
e22	v3 v9	f14 f15	2 3 21 23
e23	v0 v9	f15 f12	3 0 22 20

Vertex List

v0	0,0,0	8 9 0 23 3
v1	1,0,0	10 11 1 20 0
v2	1,1,0	12 13 2 21 1
v3	0,1,0	14 15 3 22 2
v4	0,0,1	8 15 7 19 4
v5	1,0,1	10 9 4 16 5
v6	1,1,1	12 11 5 17 6
v7	0,1,1	14 13 6 18 7
v8	.5,.5,0	16 17 18 19
v9	.5,.5,1	20 21 22 23



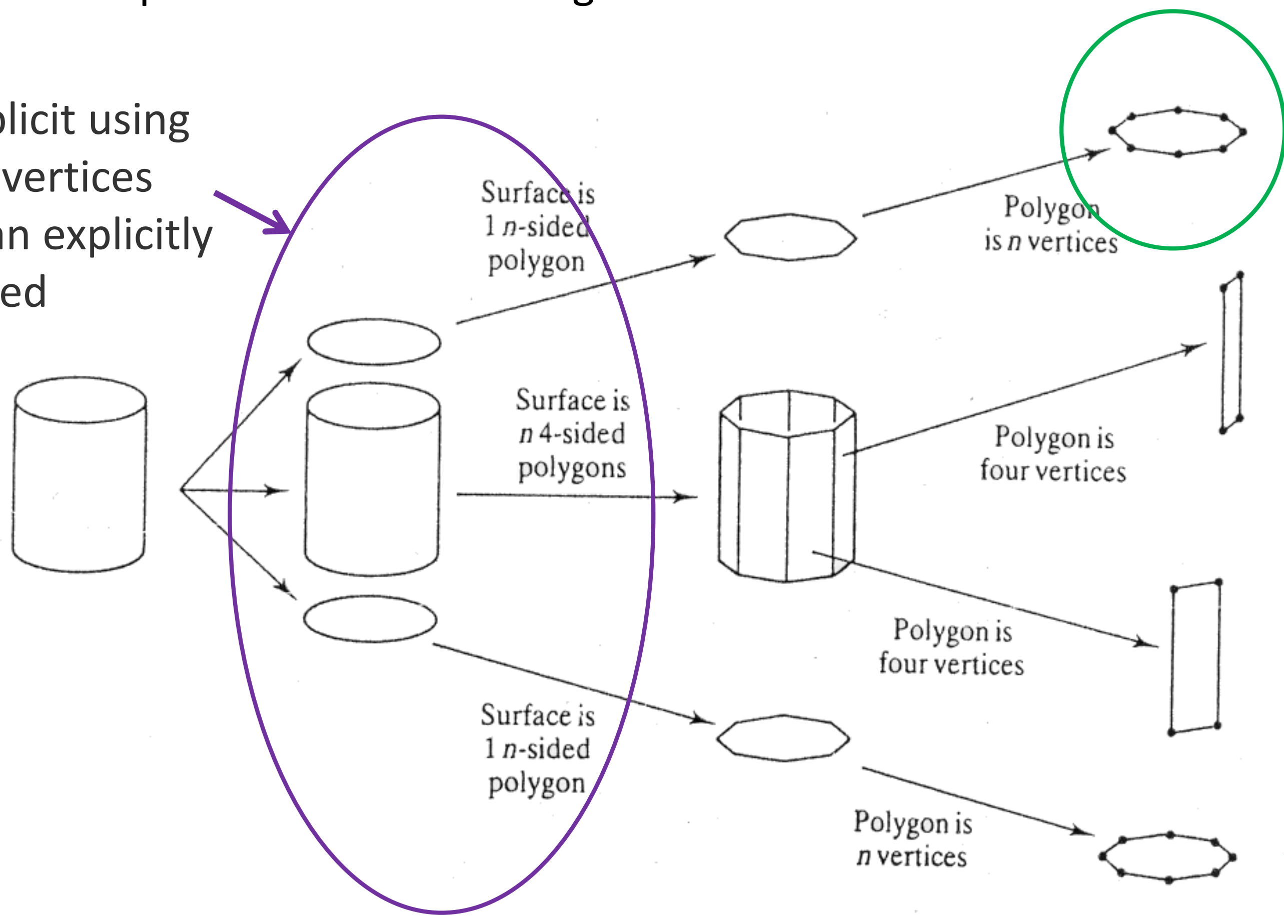
Winged Edge Structure

http://en.wikipedia.org/wiki/File:
Mesh_we2.jpg

4. A hierarchical structure

- objects → surfaces → polygons → vertices
- This structure allows ‘hard’ (or real) edges to be distinguished
  - has implications for rendering.

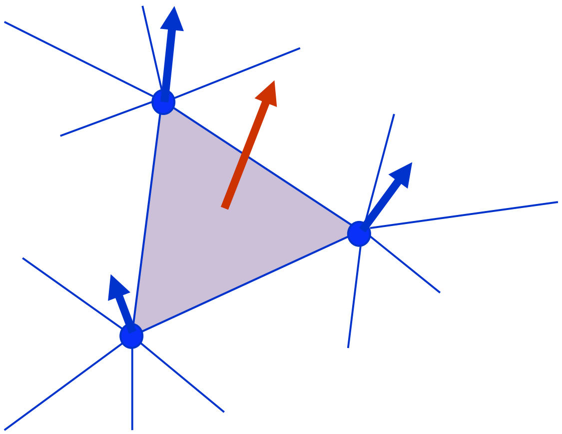
Often implicit using duplicate vertices rather than explicitly represented



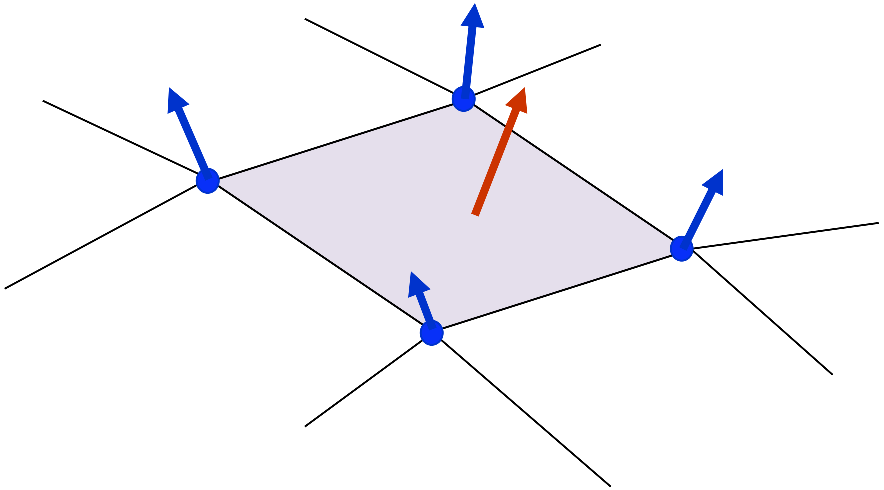
More usually split into triangles

5. Other information in the data structure

- Vertex normals
- Polygon normals
- (A normal is a vector that is perpendicular to the 'underlying' surface at that point.)
- Surface colours
- Texture coordinates, etc.



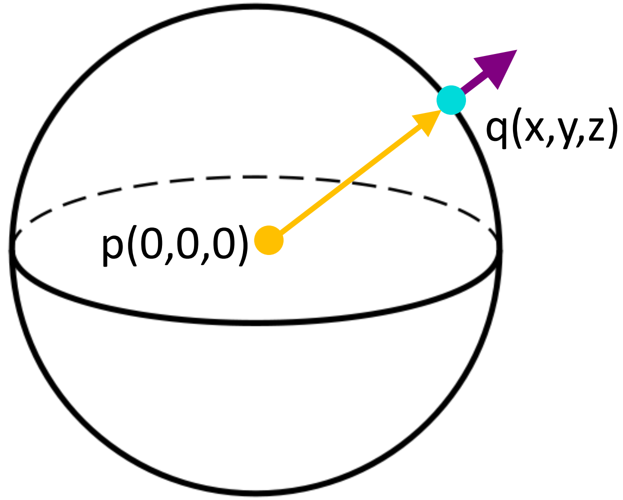
- As the data structure is processed through the graphics pipeline, other information will be added such as shading information

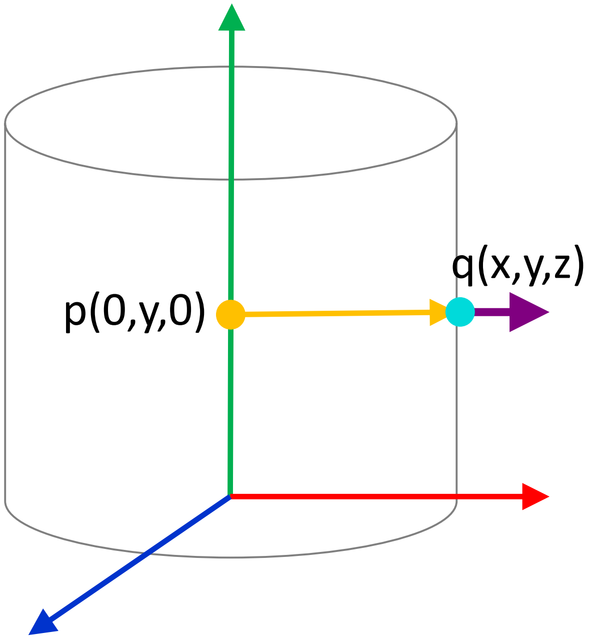


5.1 Calculating normals

Easy for certain objects:

- Flat plane defined by two of the world axes
  - Normal is the remaining world axis
- Cube aligned with world axes
  - Multiple flat planes
- Sphere
  - Use line between centre of sphere and vertex to give vertex normal q-p
- Cylinder – axis along one of the world axes
  - Use line between central axis and vertex on same plane (a circle in the plane) q-p
  - For cylinder caps (flat ends) use relevant world axis
- All normals must be normalised





5.2 Arbitrary mesh: Calculating a polygon normal

$\mathbf{V} = (x_1,y_1,z_1) - (x_0,y_0,z_0)$

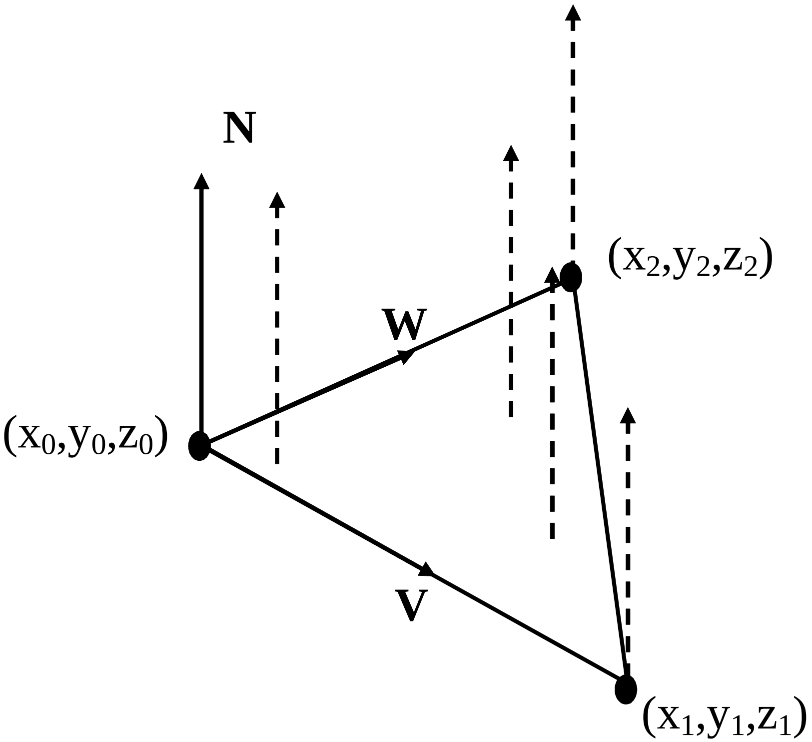
$\mathbf{W} = (x_2,y_2,z_2) - (x_0,y_0,z_0)$

$\mathbf{N} = \mathbf{V} \text{ x } \mathbf{W}$

$= (v_x,v_y,v_z) \text{ x } (w_x,w_y,w_z)$

$= ((v_y w_z - v_z w_y),$
$(v_z w_x - v_x w_z),$
$(v_x w_y - v_y w_x))$

$= (N_x, N_y, N_z)$



- Normalising:

$$\boldsymbol{N} = \left(\frac{N_x}{|\boldsymbol{N}|}, \frac{N_y}{|\boldsymbol{N}|}, \frac{N_z}{|\boldsymbol{N}|}\right)$$

where

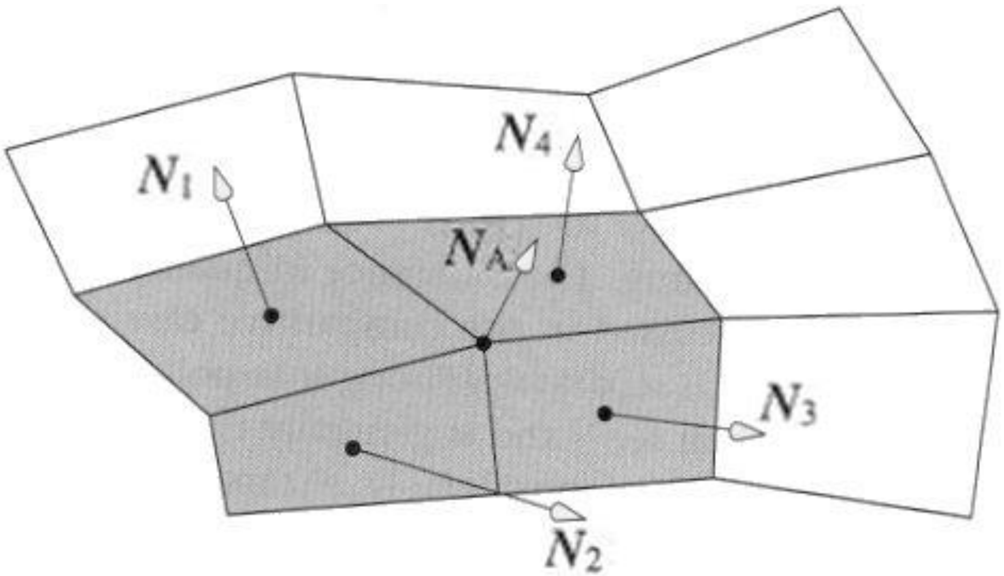
$|\mathbf{N}| = \text{sqrt}(N_x^2 + N_y^2 + N_z^2)$

Same normal all over a triangle, wherever you measure it, since the triangle is flat

5.3 Arbitrary mesh: Calculating a vertex normal

- A vertex normal (that approximates the curvature of the underlying smooth surface approximated by the polygons) can be calculated by averaging the surrounding polygon normals:

  $N_A = (N_1 + N_2 + N_3 + N_4)/4$



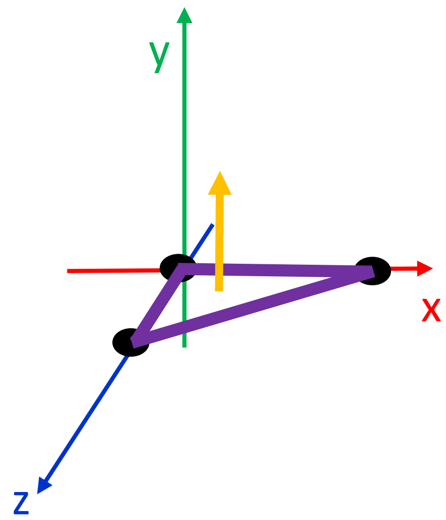
- Then normalise: $N_A = N_A/|N_A|$

Question

- Given a triangle with vertex 1 at position (0,0,0), vertex 2 at position (0,0,2) and vertex 3 at position (3,0,0), with vertices ordered in an anticlockwise direction, write down the triangle normal in its normalised form.

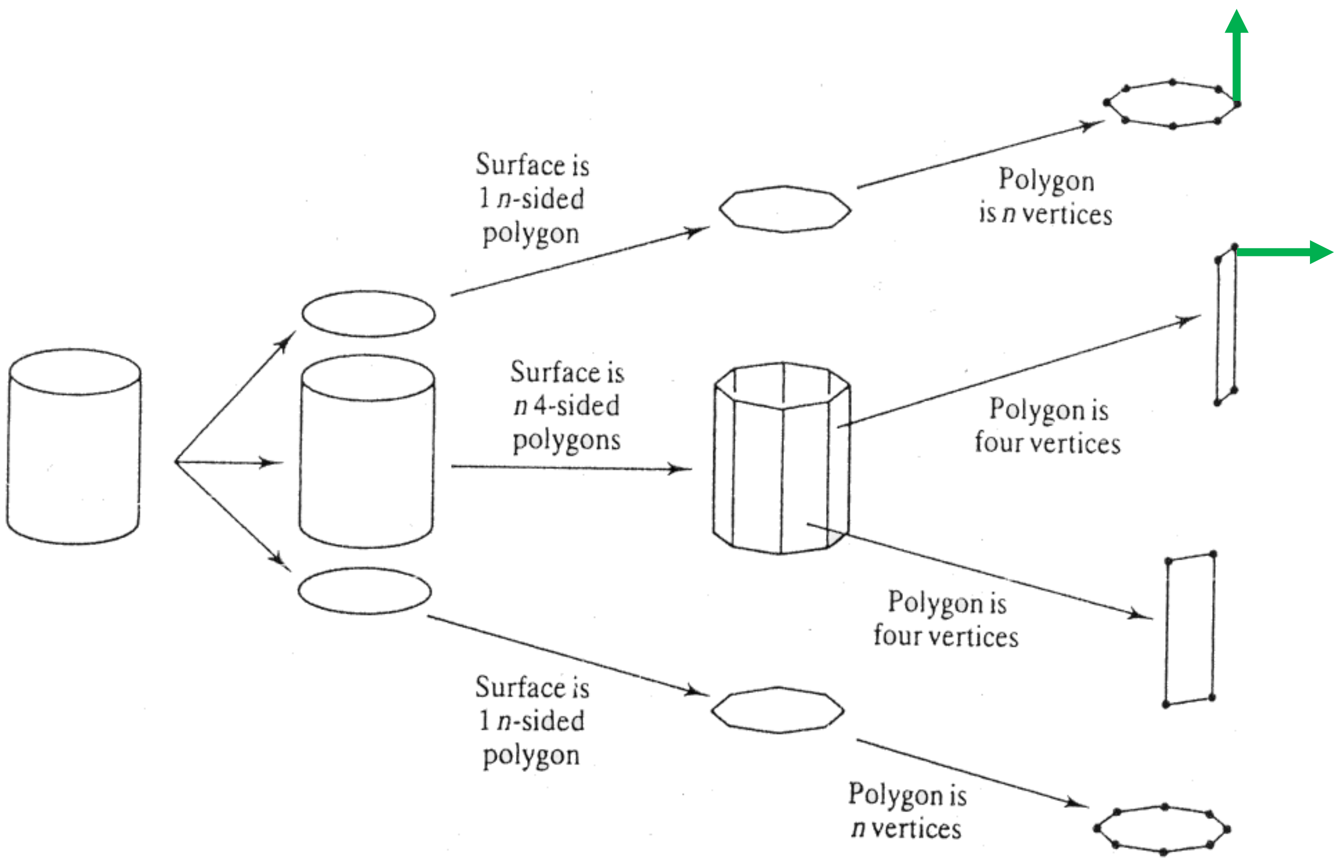
Question

- Given a triangle with vertex 1 at position (0,0,0), vertex 2 at position (0,0,2) and vertex 3 at position (3,0,0), with vertices ordered in an anticlockwise direction, write down the triangle normal in its normalised form.



5.4 Hard edges

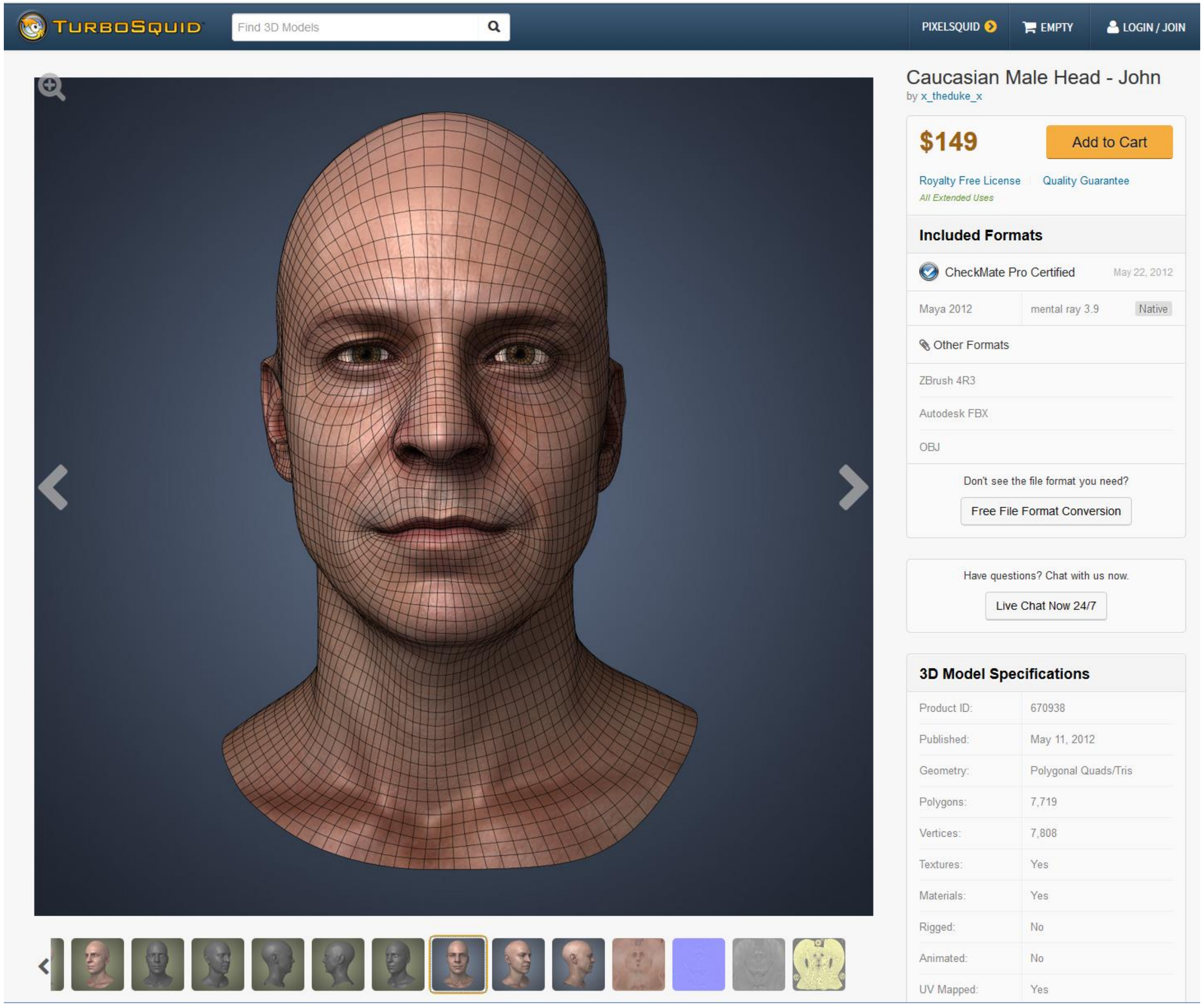
- Multiple normals for a vertex on a hard edge
- *A solution:* Duplicate vertices – separate surfaces may be explicitly represented in the data structure



6. Techniques for creating polygonal objects

Lots of techniques.
Here's a few:

- Purchase
- Scanning
- Modelling software
- Mathematical generation
- Sweeping
- Procedural techniques



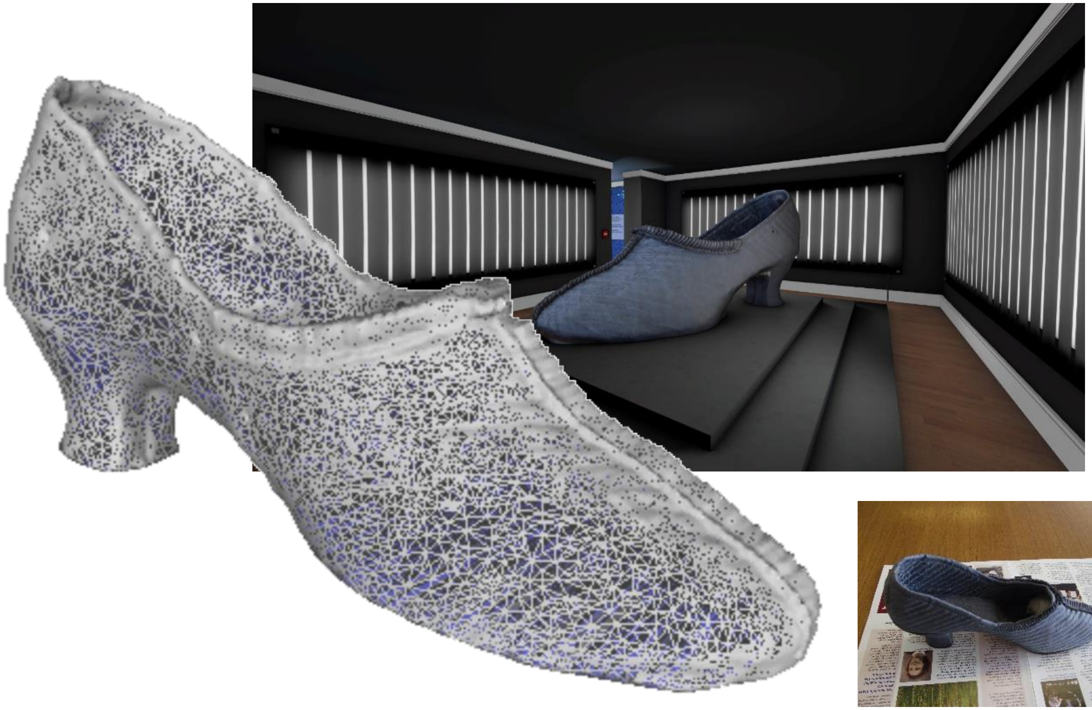
www.turbosquid.com

6.1 Digital acquisition of shape

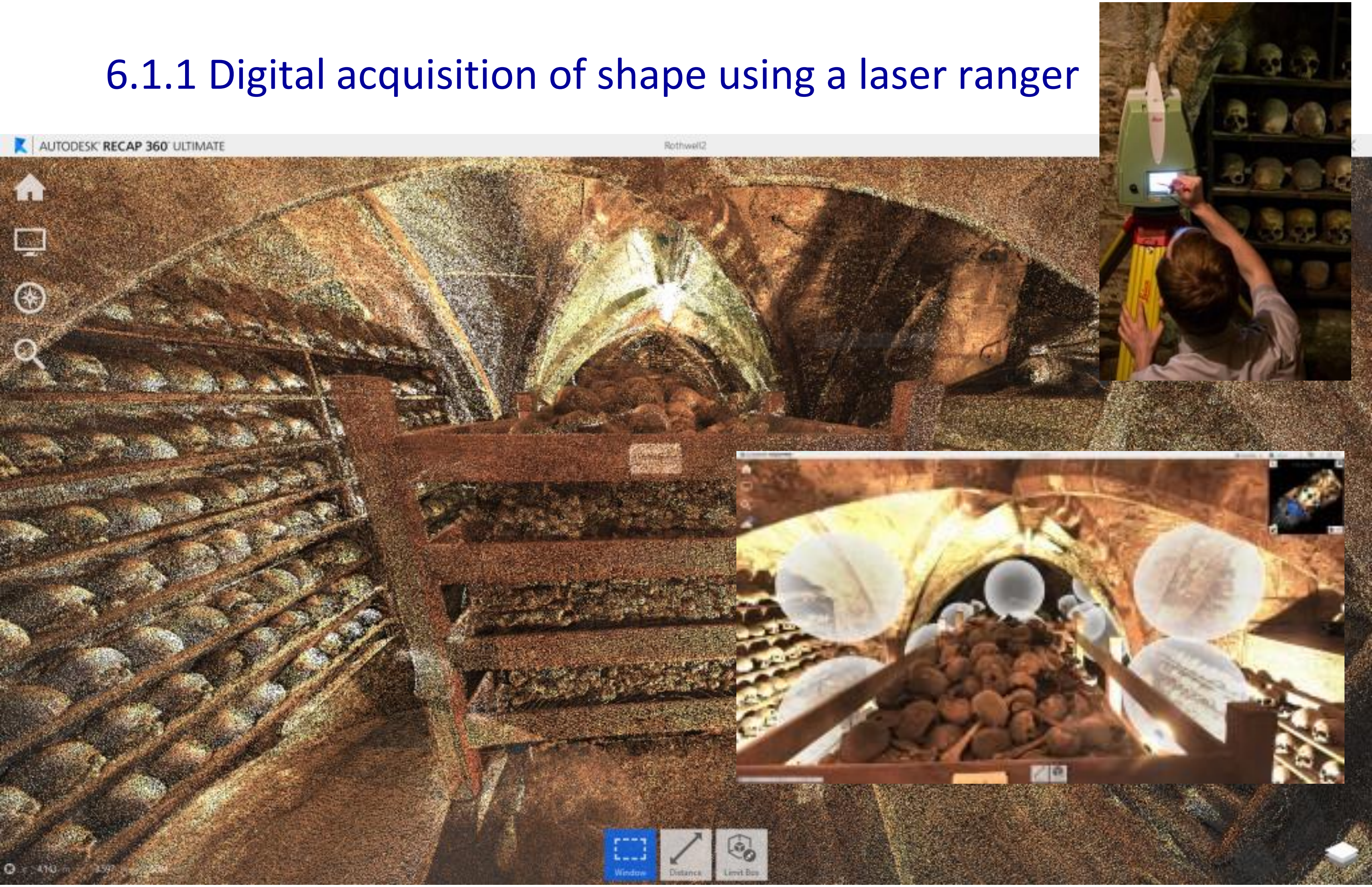
- 3D digitiser – manual
- Laser ranger – automatic
- Multiple photographs, e.g. Autodesk 123D Catch

http://www.3d-microscribe.com

http://www.artec3d.com/hardware/artec-spider/



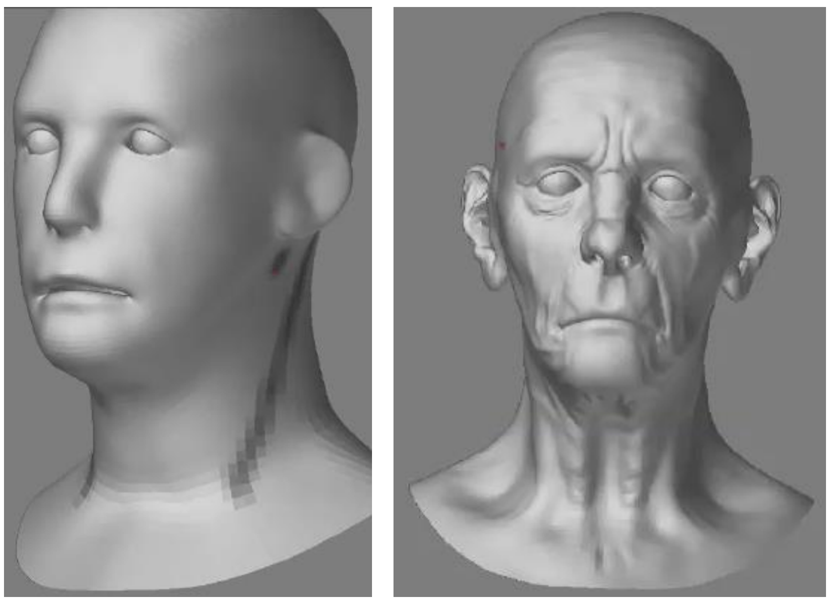
6.1.1 Digital acquisition of shape using a laser ranger

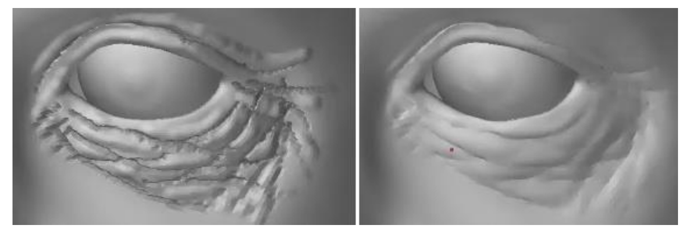


http://www.rothwellcharnelchapel.group.shef.ac.uk/

6.2 Modelling software

- Mesh editing, e.g. 3ds Max, Maya, Blender, …
- Sculpting, e.g. Mudbox, ZBrush

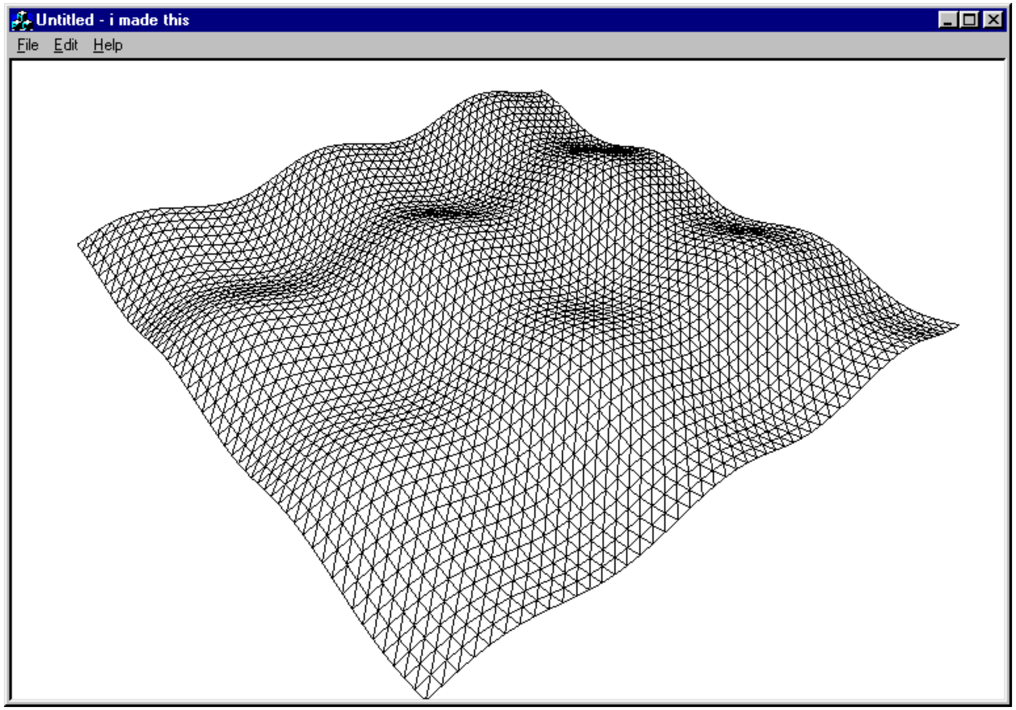




Zbrush, www.pixologic.com

6.3 Mathematical description

- Discretise the surface described by a function
- Example: $z = \sin x + \cos y$

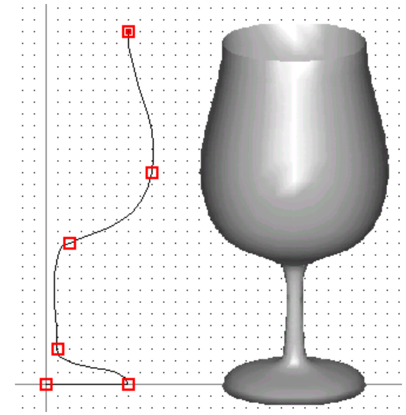


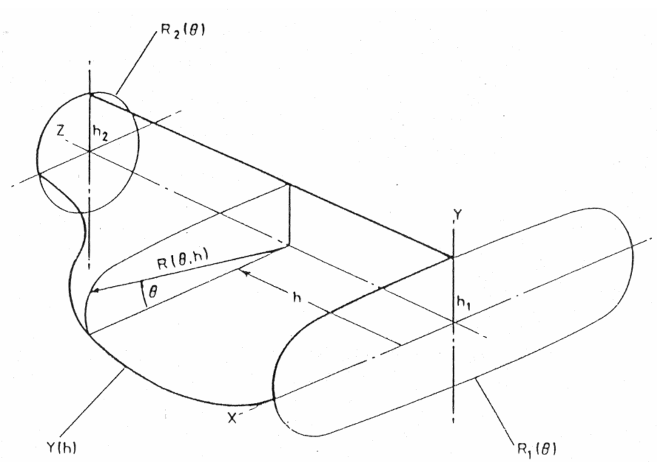
Wire-frame view of a 50 x 50 height map, giving 4802 triangles.

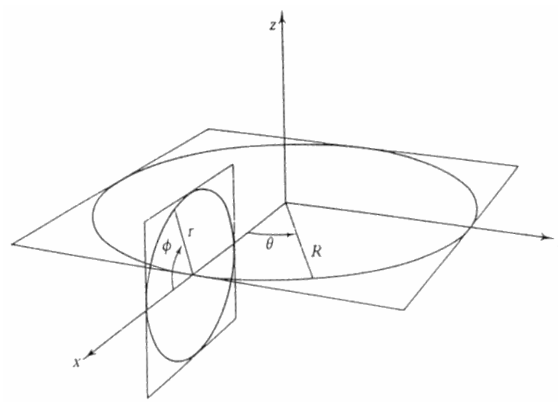
6.4 Sweeping

Alternatives:

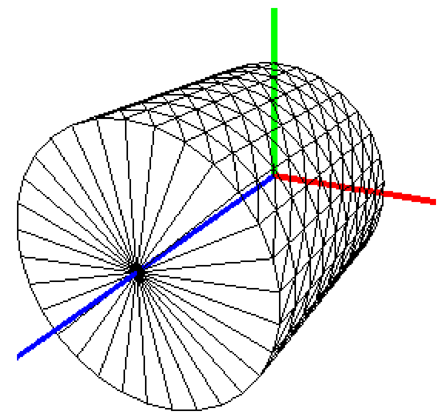
- Rotational sweep or surface of revolution;
- Translational sweep or extrusion;
- Ducted solids or generalised cylinders.





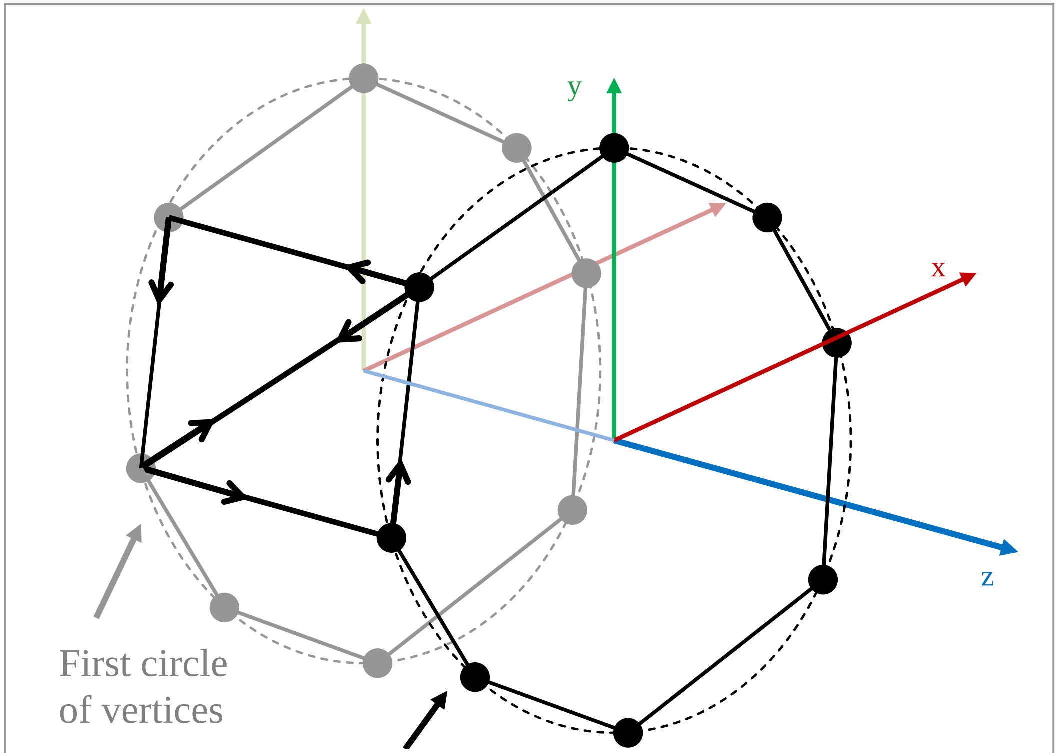


Parametric scheme for generating a wireframe toroid.

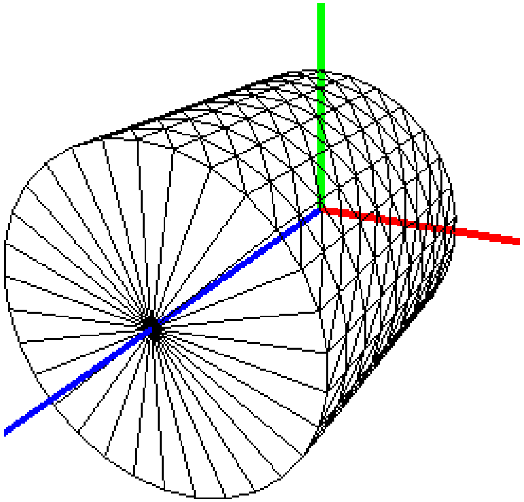


6.4 Sweeping

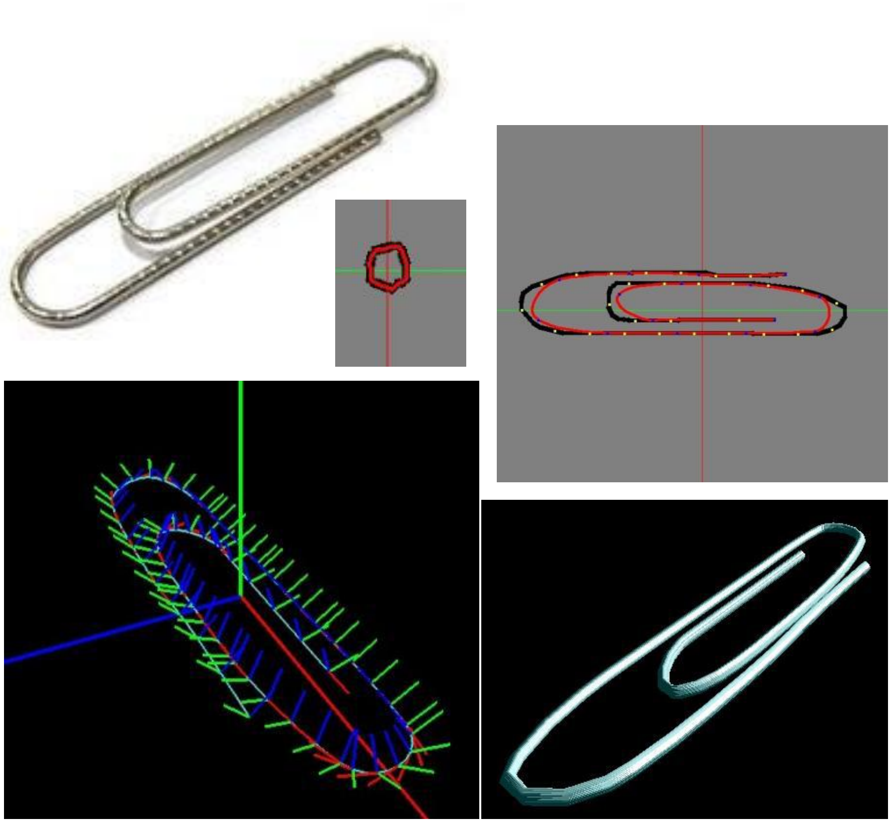
- Translational sweep or extrusion;

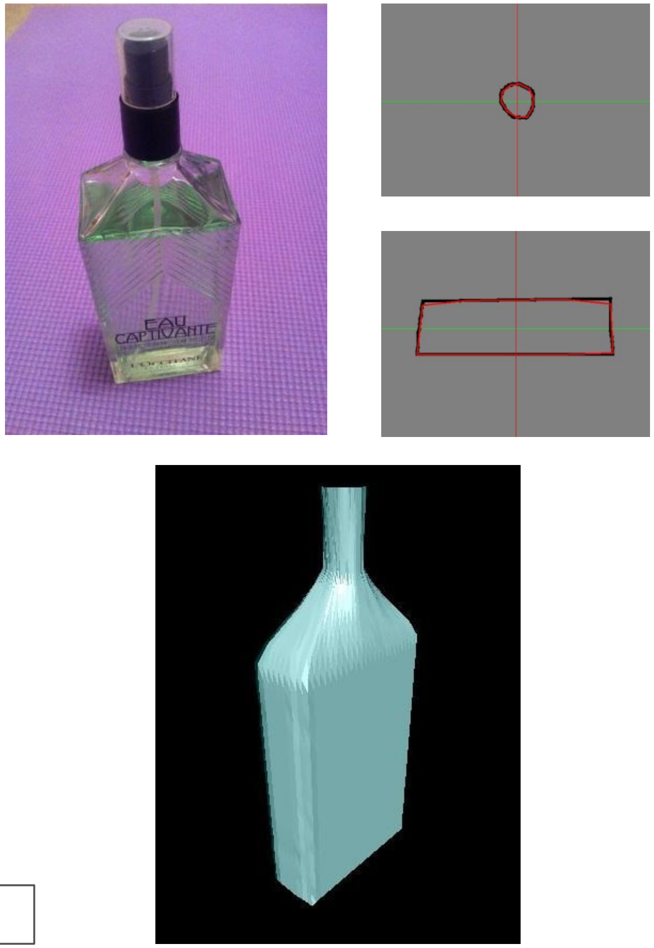


Subsequent circles of vertices are attached to previous circles to create triangles. The arrows on triangle edges show winding order.



6.4.1 Examples

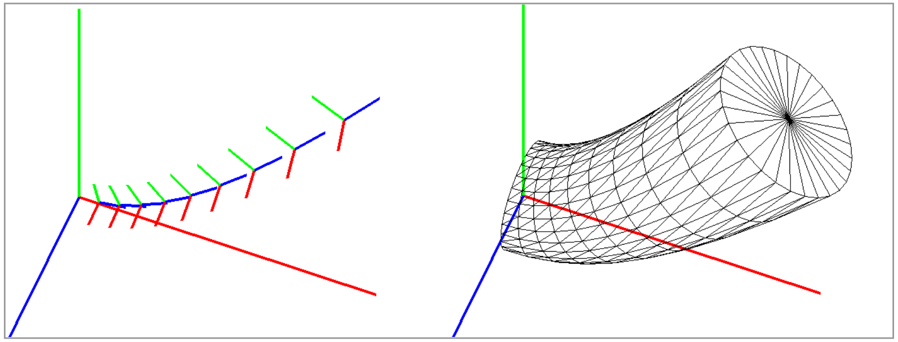


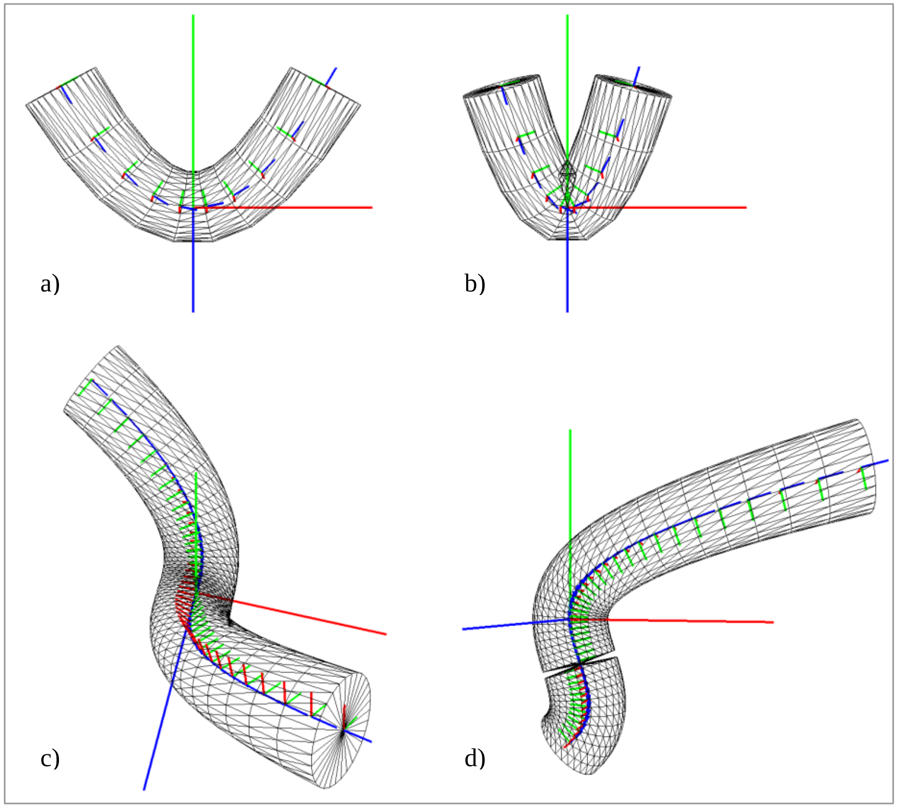


Yi-Kan Chen, MSc dissertation, 2012

6.4.2 Issues

- Arc length parameterisation;
- Curvature;
- Reference frames, e.g. Frenet frames.





Frenet frames:
Jules Bloomenthal, "Calculation of Reference Frames along a Space Curve", in Graphics Gems, Academic Press, 1990. pp. 567-571
http://webhome.cs.uvic.ca/~blob/courses/305/notes/pdf/ref-frames.pdf
W. Wang, B. Jüttler, D. Zheng, and Y. Liu, "Computation of rotation minimizing frames", ACM Transactions on Graphics (TOG), Volume 27 Issue 1, March 2008

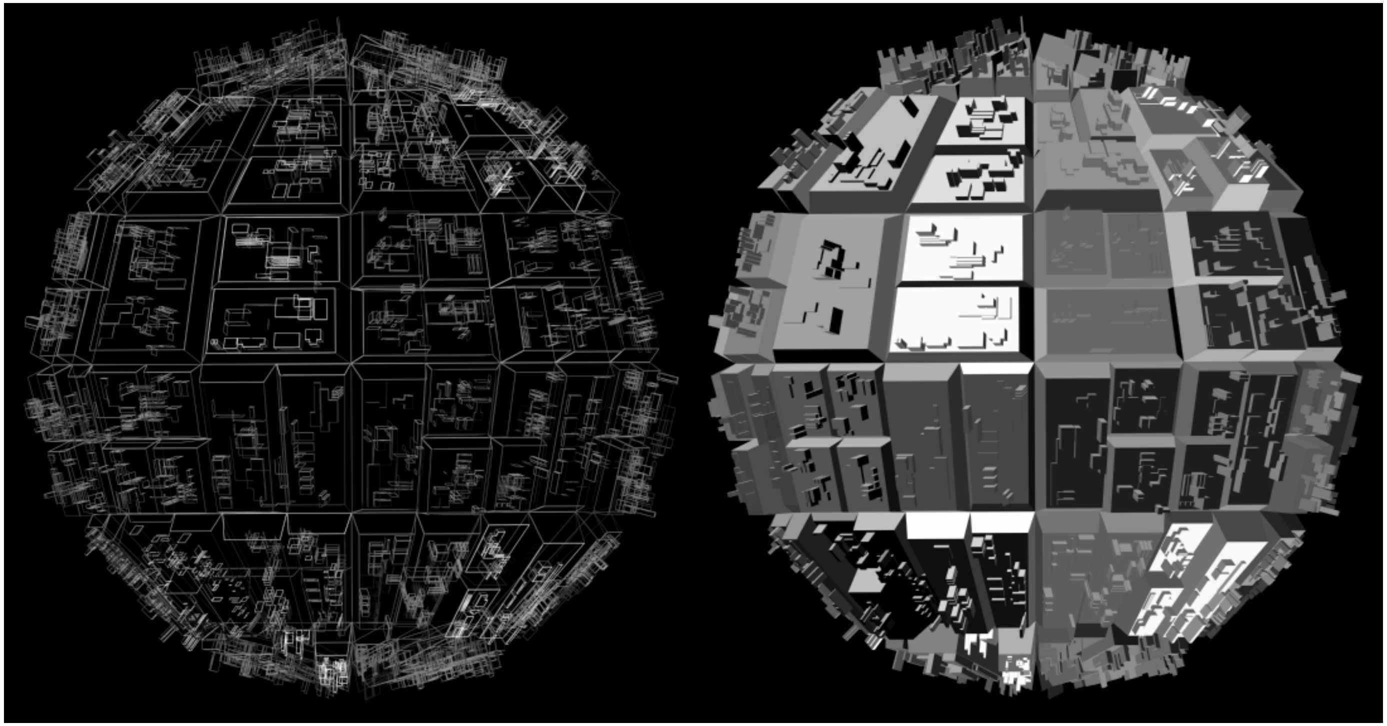
6.4.3 Sweeping in a photo…

- Tao Chen, Zhe Zhu, Ariel Shamir, Shi-Min Hu, Daniel Cohen-Or
  3-Sweep: Extracting Editable Objects from a Single Photo
  Proc. Siggraph Asia 2013

https://www.youtube.com/watch?v=Oie1ZXWceqM

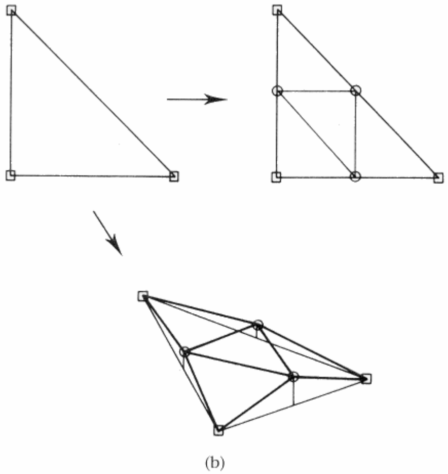
6.5 Procedural techniques

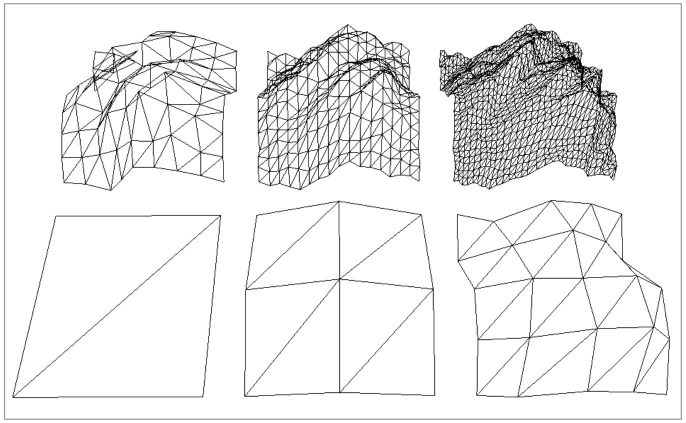
- Parameterize surface attributes to create variations on a model.
- *Example*: A truncated pyramid and a cuboid are stochastically distributed on the surface of a sphere to produce a more complex object (produced using a 3ds Max plug-in).

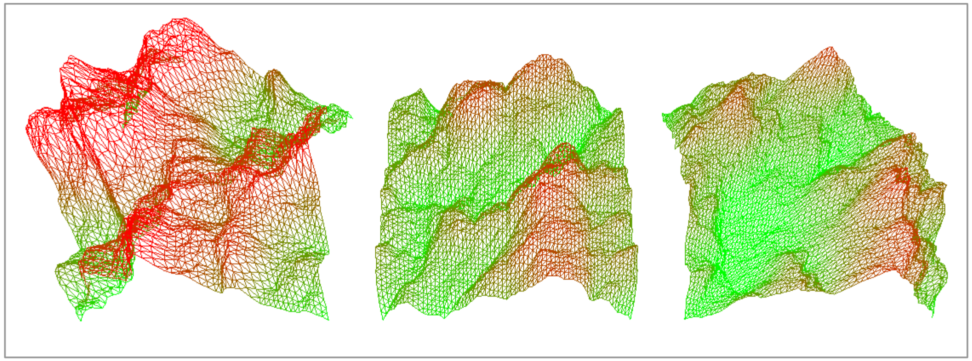


6.6 Procedural technique: Fractal objects

- Recursively subdivide each edge, generating a displacement in a direction normal to the plane of the original facet.





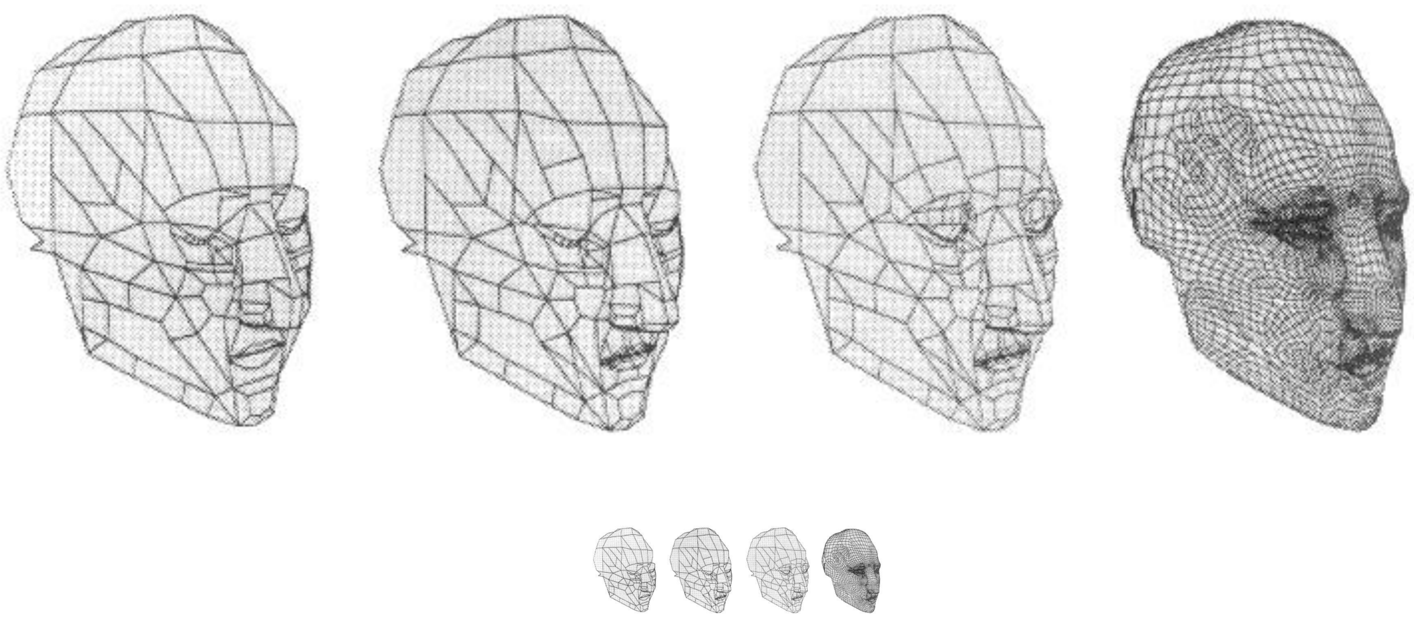


Gamito, 2006

Further details: Fournier, A., D.Fussell and L.Carpenter (1982). Computer rendering of stochastic models. Comm. ACM, 25(6), pp.371-84

7. Level-of-detail (LOD)

- For a complex object, large numbers of polygons are needed to capture the detail
- If such an object projects onto a small area of the screen, effort is wasted
- A solution is to use a Level of Detail (LOD) approach:
  - A series of models, each with successively less polygons



7.1 Progressive meshes (Hoppe, 1996)

http://research.microsoft.com/~hoppe/

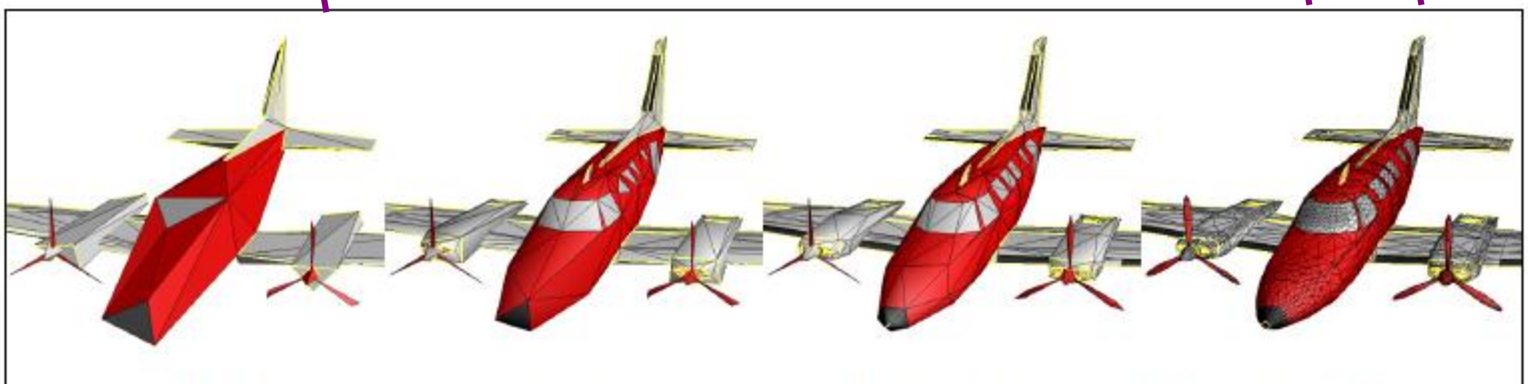
8. Summary

- Polygons are ubiquitous:
  - Can model any complex object
  - Efficient rendering
- Lots of alternative data structures
  - Face-vertex is common
- Level of Detail approach to choose mesh based on viewing distance
  - (Further info: Luebke et al, Level of Detail for 3D Graphics, Morgan Kaufmann, 2003 (http://lodbook.com/)
  - Special techniques for terrain (http://www.vterrain.org)
- Many ways of producing polygon models from scanning to procedural generation

A.1 Progressive meshes (Hoppe, 1996)

- Mesh optimisation techniques on highest detail layer M^n to construct lower layers

...



(a) Base mesh M^0 (150 faces) (b) Mesh M^{175} (500 faces) (c) Mesh M^{425} (1,000 faces) (d) Original $\hat{M}=M^n$ (13,546 faces)

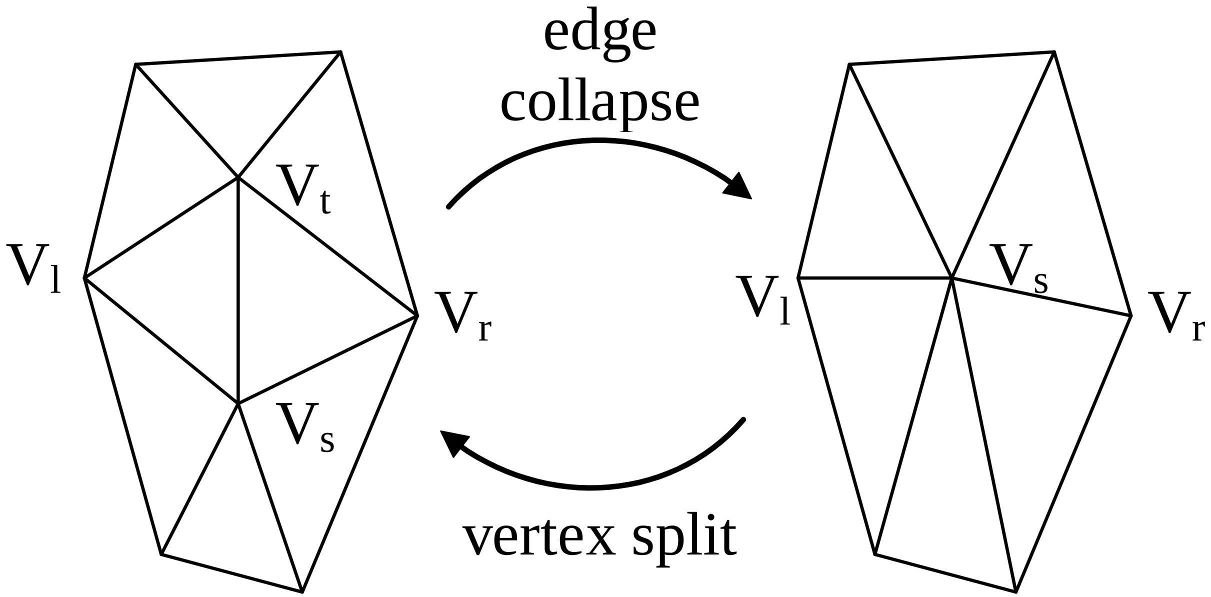
Figure 5: The PM representation of an arbitrary mesh $\hat{M}$ captures a continuous-resolution family of approximating meshes $M^0 \ldots M^n=\hat{M}$.

M^0 M^{175} M^{423} M^n

Hoppe, H. (1996). Progressive meshes. Proc SIGGRAPH'96. pp.99-108.

A.1 Progressive meshes (Hoppe, 1996)

- Edge collapse operation: Two vertices combined into one
- Store coarsest level of detail M_0 and information to ascend from layer to layer to highest detail layer M_n
- (progressive meshes were added in DirectX 8)



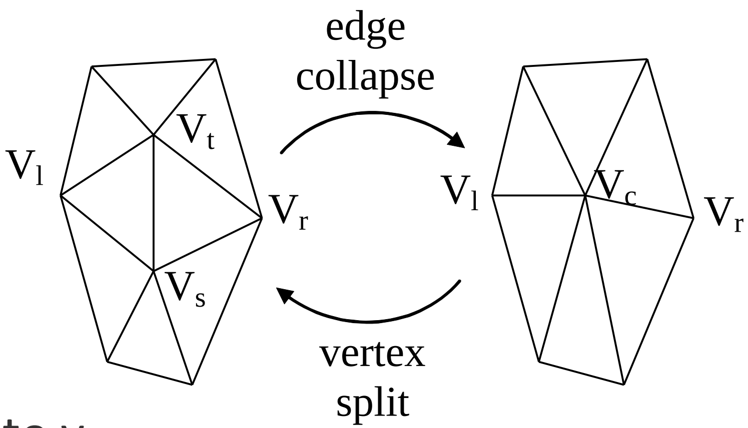
Hoppe, H. (1996). Progressive meshes. Proc SIGGRAPH'96. pp.99-108.

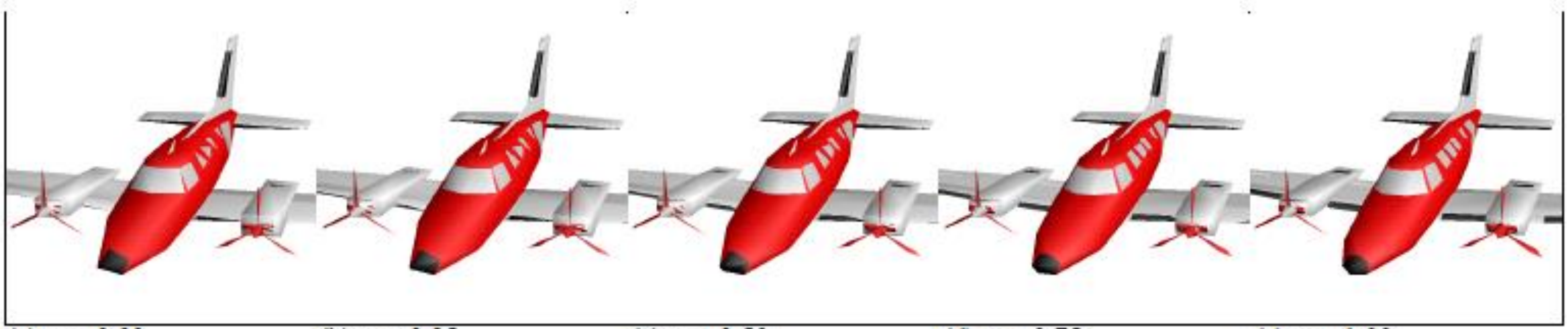
A.1 Progressive meshes (Hoppe, 1996)

- 'geomorphs', which are geometric blends in 3D space

$$V_c \in \left| V_t , V_s , \frac{V_t + V_s}{2} \right|$$

- Linear interpolation of v_c to v_t and also to v_s





(a) $\alpha = 0.00$ (b) $\alpha = 0.25$ (c) $\alpha = 0.50$ (d) $\alpha = 0.75$ (e) $\alpha = 1.00$

Figure 6: Example of a geomorph $M^G(\alpha)$ defined between $M^G(0)=M^{175}$ (with 500 faces) and $M^G(1)=M^{425}$ (with 1,000 faces).

Hoppe, H. (1996). Progressive meshes. Proc SIGGRAPH'96. pp.99-108.

A.2 A simple criterion for edge collapse

- Hoppe uses a fairly complex collapse criterion based on minimising an energy function over the mesh
- A simple metric that can be used to decide the edge for collapse is:

$$\frac{|V_s - V_t|}{|\mathbf{N}_s \cdot \mathbf{N}_t|}$$

which is the length of an edge (V_s-V_t) divided by the dot product of its two vertex normals

- When it is continually applied the mesh will reach a point where it begins to 'collapse'

B. Data transfer between CPU and GPU

- Draw each triangle separately – inefficient
- Triangle strip – compress connectivity information
- Array of vertices – transfer arrays of information
- Display list – used with fixed function pipeline
  - Transfer data to GPU once
  - Single call to redraw the list stored on the GPU
  - If edit vertices then re-send all data
- Vertex Buffer Objects – commonly used when using the programmable pipeline
  - Advantages of both vertex array and display list
  - Store on GPU, but can edit the individual vertices
  - Can use separate buffers for vertex data and index data

