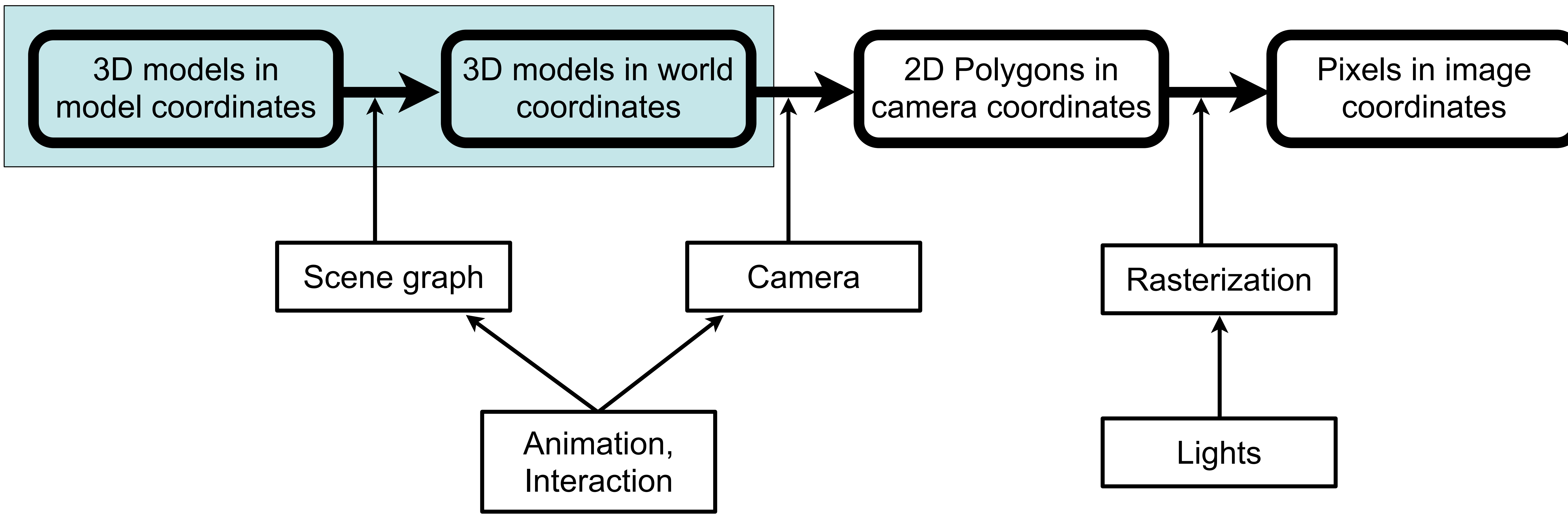
Computer Graphics 1

Chapter 2 (April 29th, 2010, 2-5pm):

3D models and their descriptions

The 3D rendering pipeline (our version for this class)

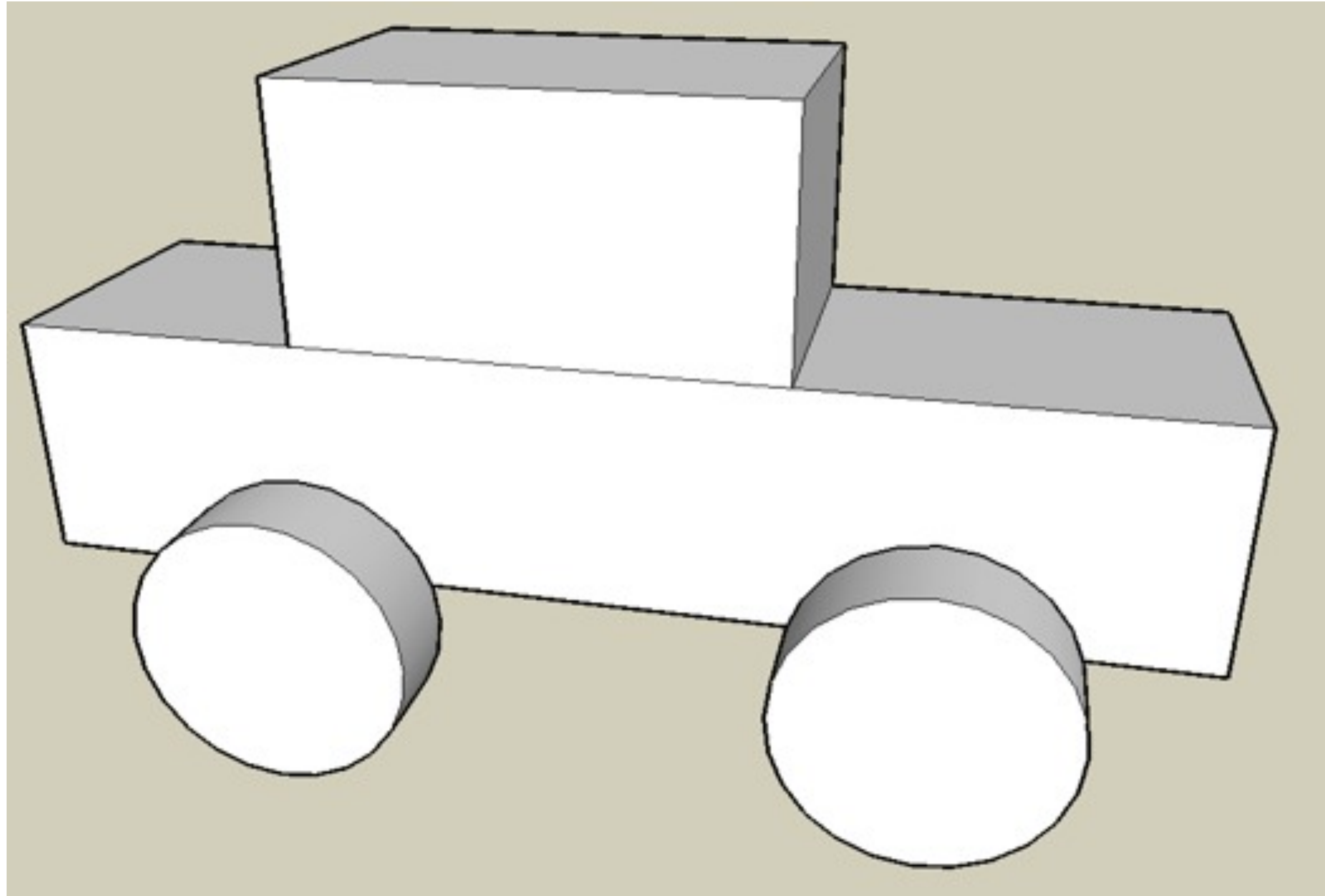


Geometric Primitives

- Simplest way to describe geometric objects
- Can be used directly by some renderers (e.g., Ray tracing)
- Can be transformed into polygons easily (Tesselation)
- Can be transformed into Voxels easily
- Useful for creating simple block world models

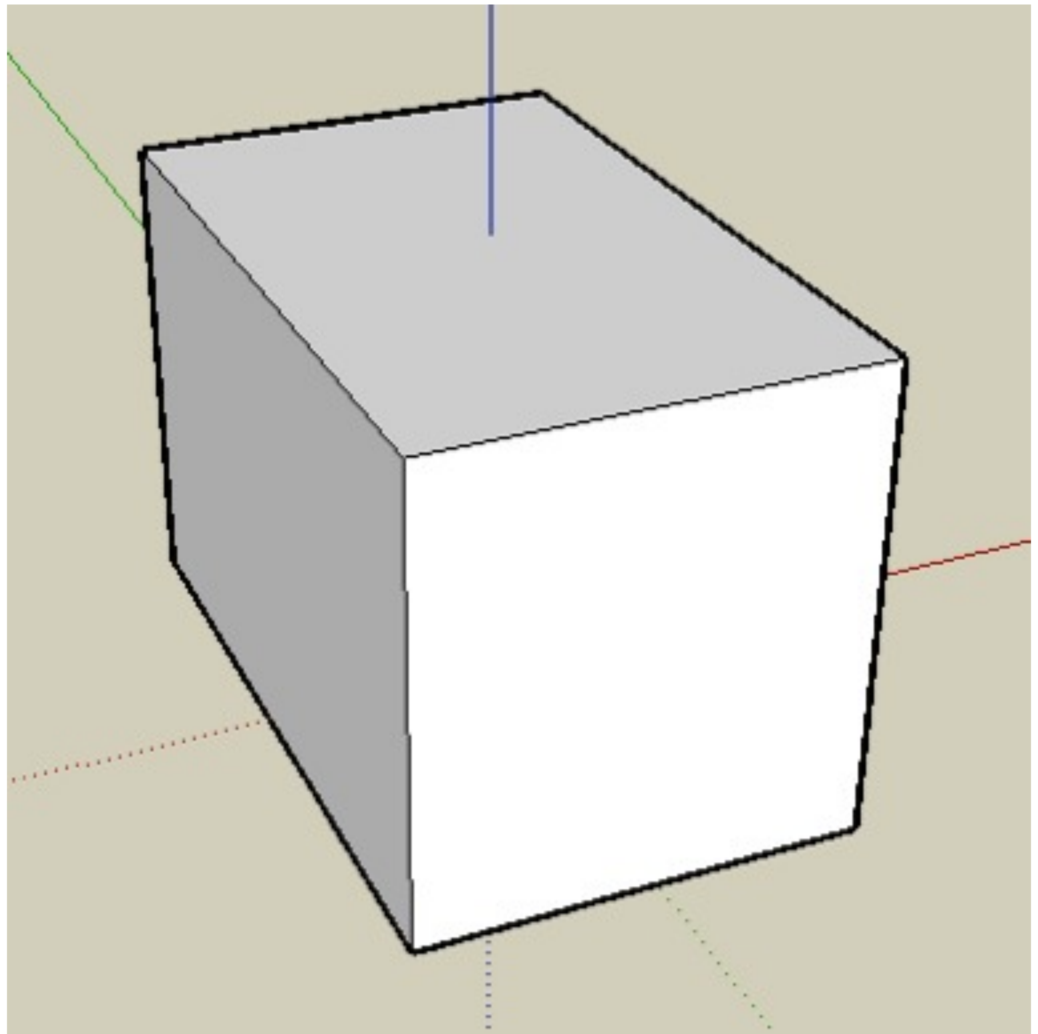
- Good start for modeling in VRML/X3D

- Objects can intersect/penetrate



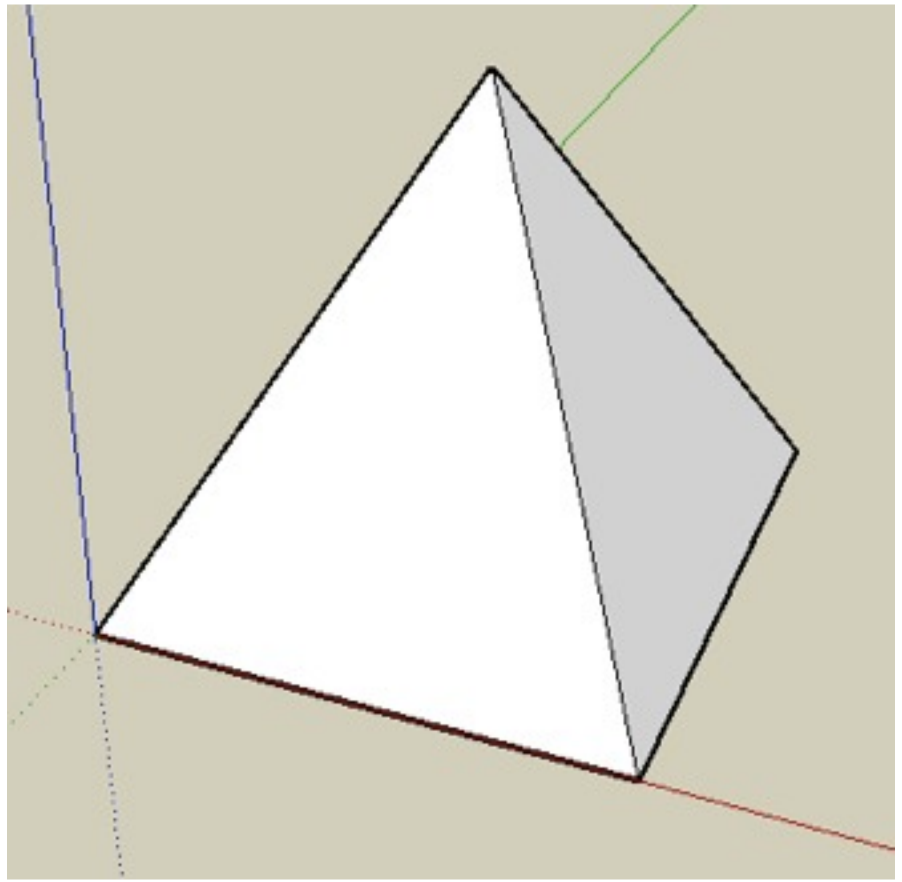
Box

- Described by (width, length, height)
- Origin usually in the center
- 8 points, 12 edges, 6 rectangles, 12 triangles

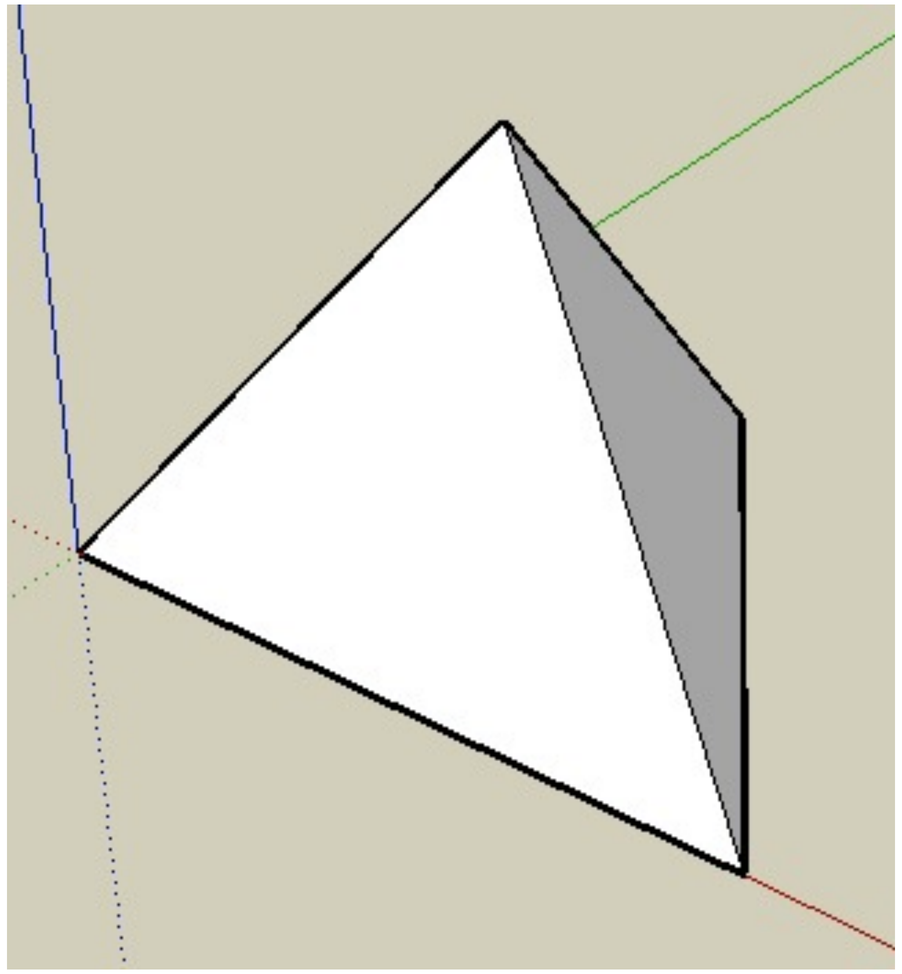


Pyramid, Tetrahedron

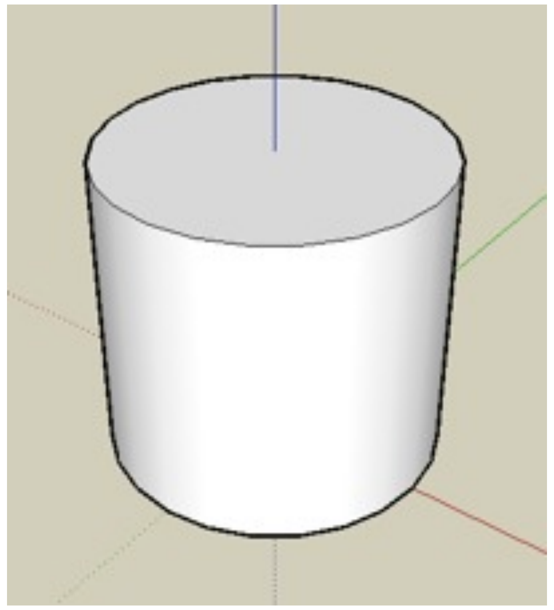
- Basis of pyramid = rectangle
- given by (width, length, height)
- 5 points, 8 edges, 6 triangles



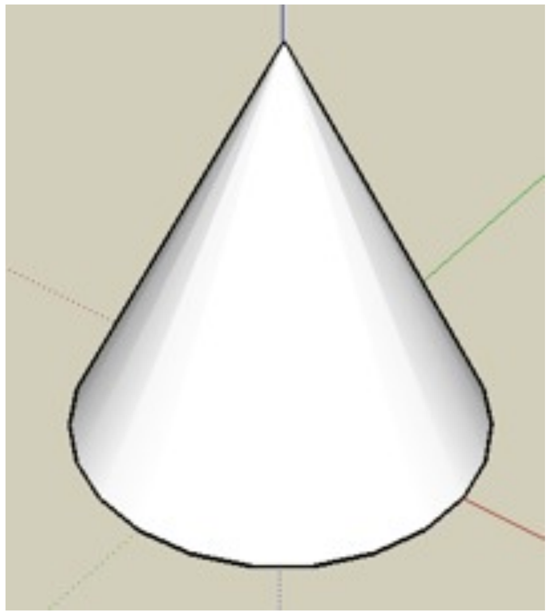
- Basis of tetrahedron = triangle
- given by (width, length, height)
- 4 points, 6 edges, 4 triangles,



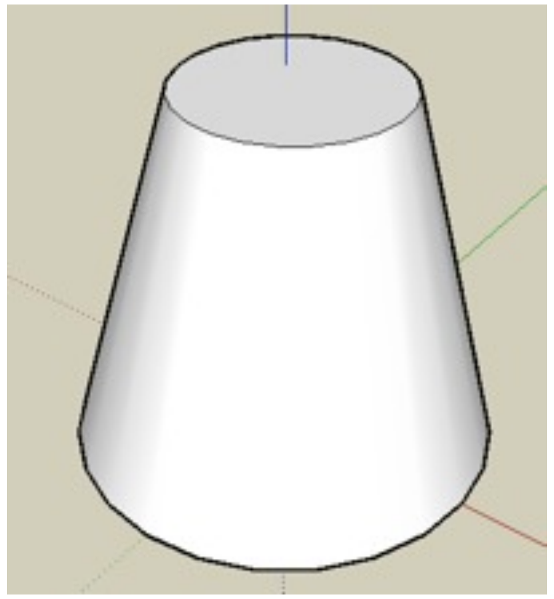
Cylinder, cone, truncated cone



- Cylinder given by (radius, height)
- Number of polygons dep. on tesselation



- Cone given by (radius, height)
- Number of polygons dep. on tesselation

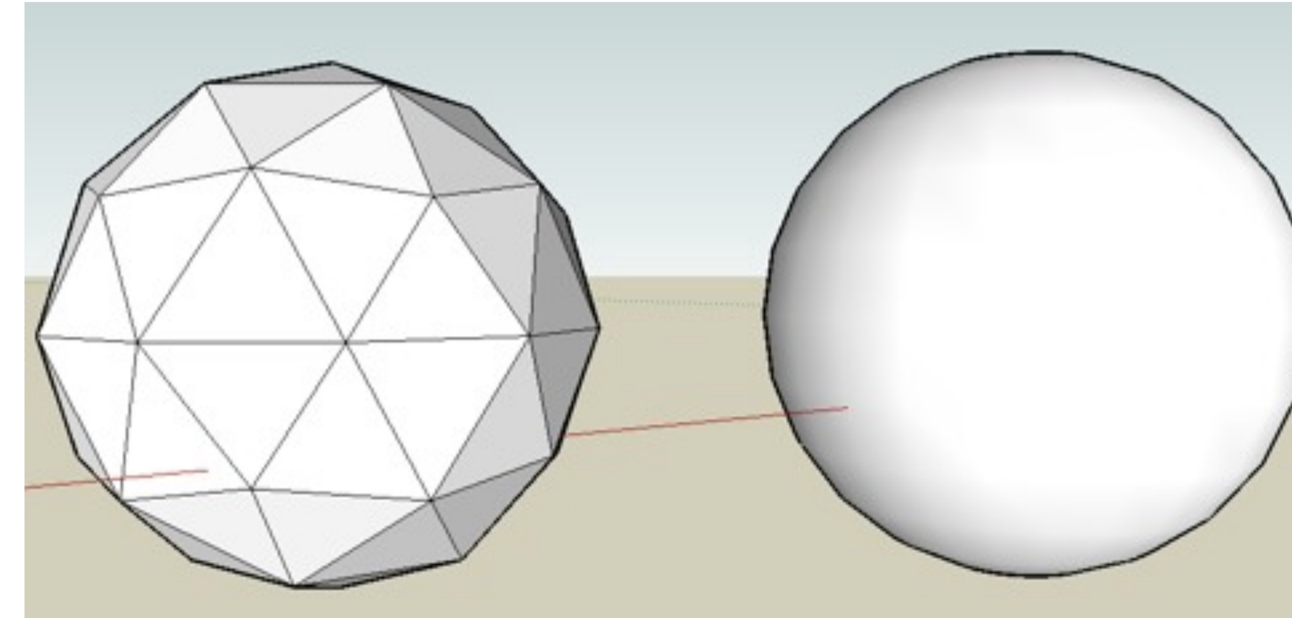


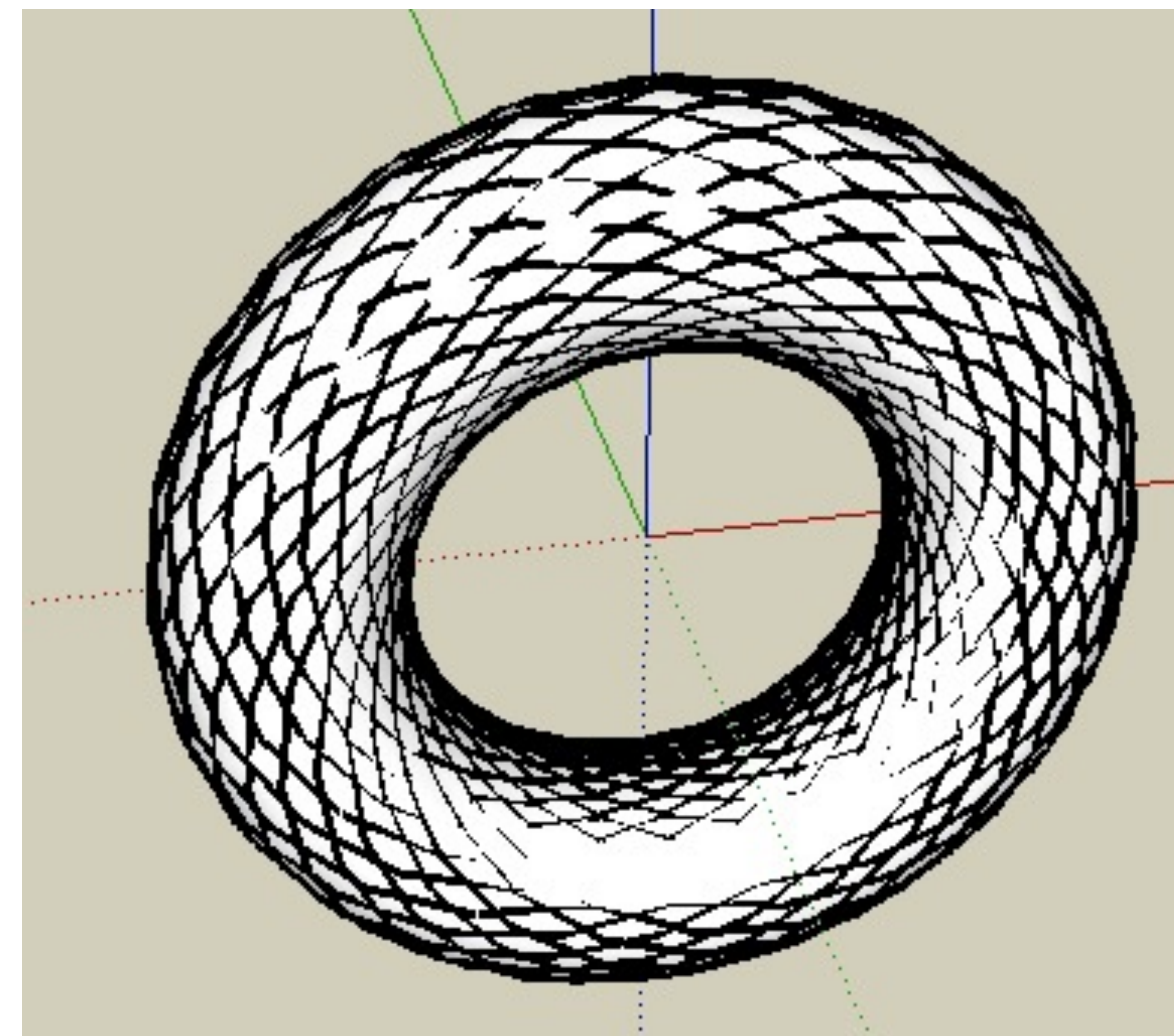
- Truncated cone given by (r1, r2, height)
- Number of polygons dep. on tesselation

- Which of these would you rather have if you only had one available?

Sphere, Torus

- Sphere is described by (radius)
- Torus is defined by (radius1, radius2)
- Number of polygons dep. on tesselation





Geometric Primitives: Summary

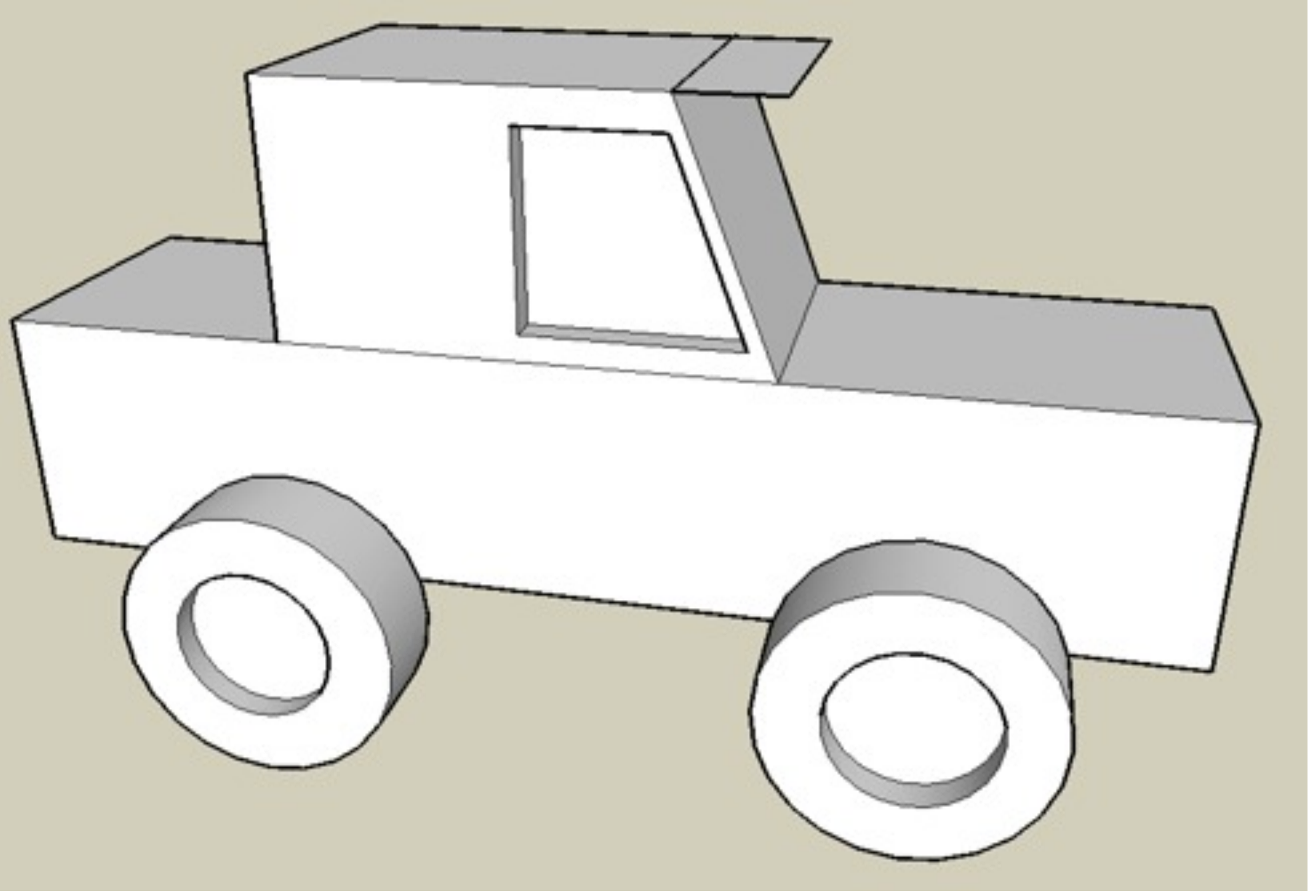
- Not all of these exist in all graphics packages
- Some packages define additional primitives (dodecahedron, teapot...;-)

- Practically the only way to model VRML or X3D in a text editor
- Can give quite accurate models
- Extremely lean! very few polygons

- Think of application areas even in times of powerful PC graphics cards!
  –
  –
  –
  –

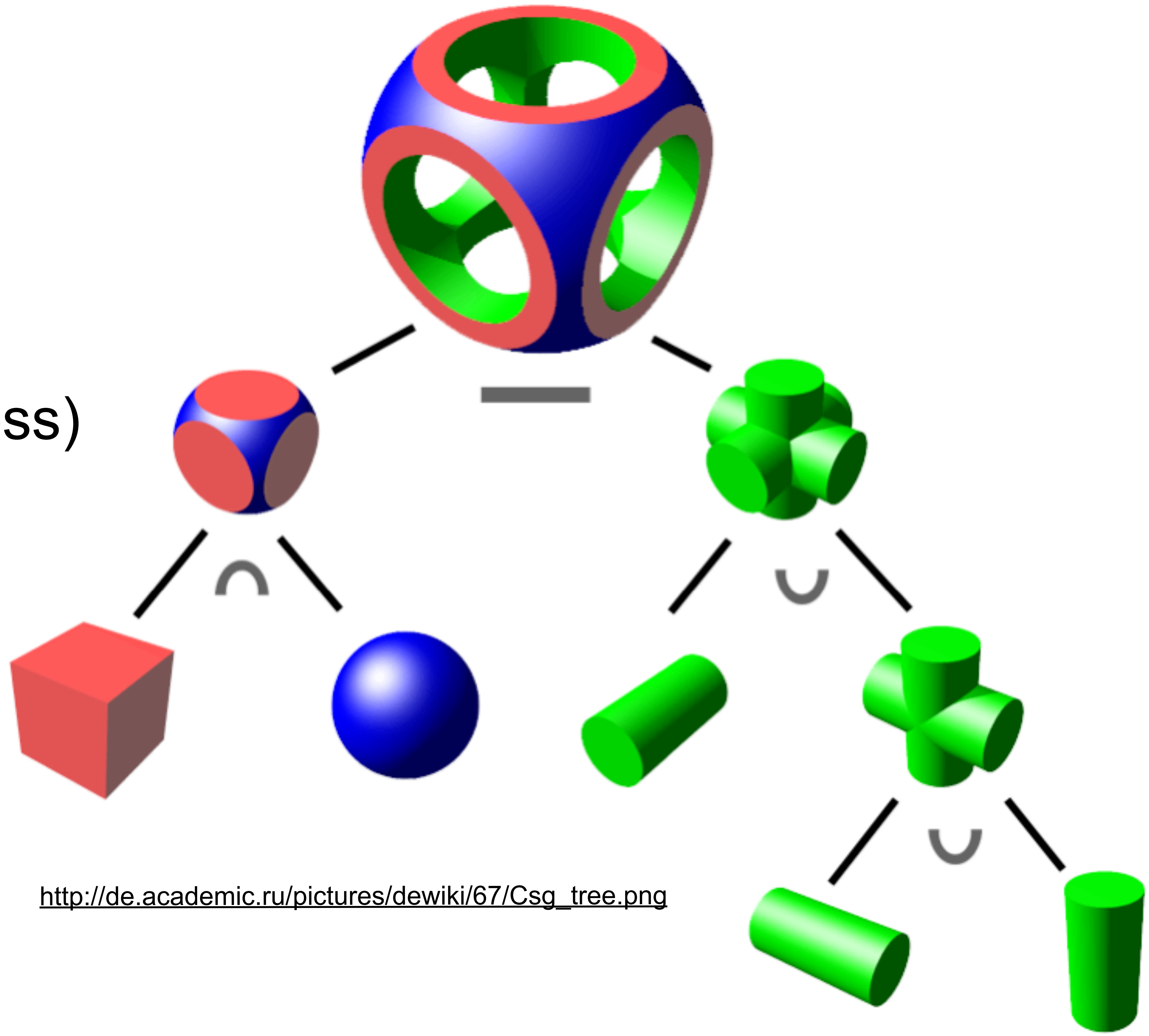
Constructive Solid Geometry

- Basic idea: allow geometric primitives and all sorts of boolean operations for combining them
- Can build surprisingly complex objects
- Good for objects with holes (often the simplest way)
- Basic operations:
  - **Or**: combine the volume of 2 objects
  - **And**: intersect the volume of 2 objects
  - **Not**: all but the volume of an object
  - **Xor**: all space where 1 object is, but not both
- Think about:
  - wheels of this car
  - tea mug
  - coke bottle (Problems??)



CSG: a complex Example

- rounded_cube =
  cube **And** sphere
- cross =
  cyl1 **Or** cyl2 **Or** cyl3
- result =
  rounded_cube **And** (**Not** cross)

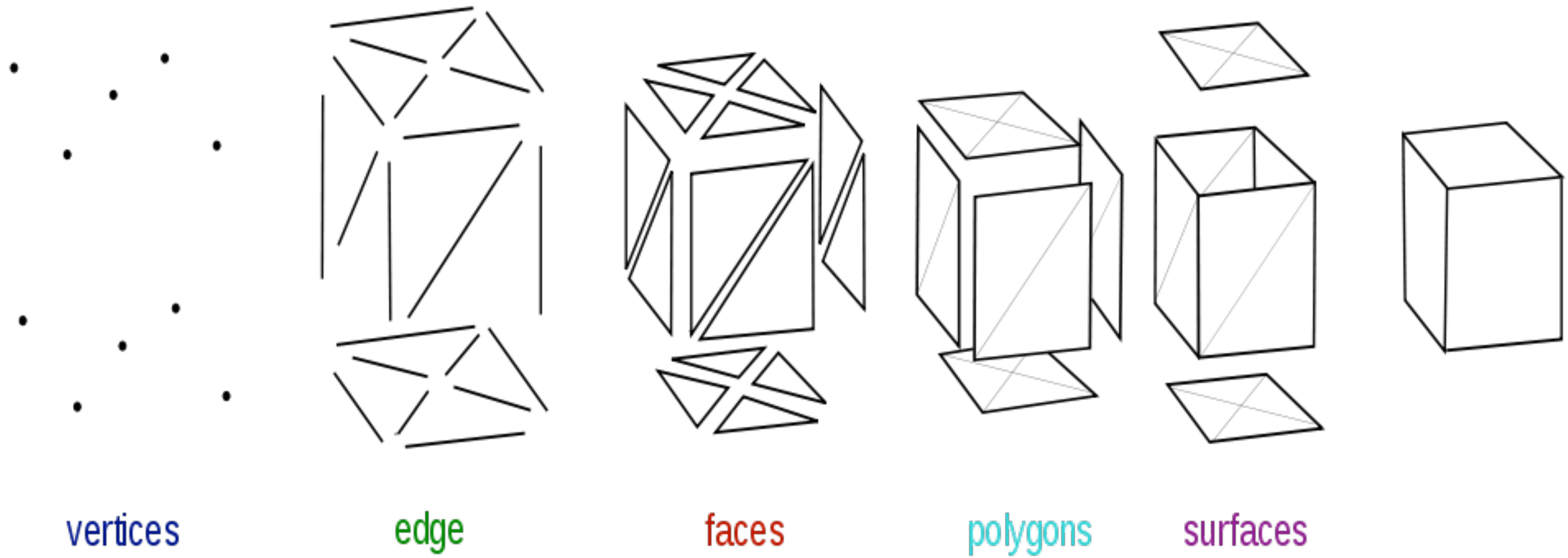


http://de.academic.ru/pictures/dewiki/67/Csg_tree.png

- Think: Are CSG operations associative?
  –
- ...commutative?
  –

Polygon Meshes

- Describe the surface of an object as a set of polygons
- Mostly use triangles, since they are trivially convex and flat
- Current graphics hardware is optimized for triangle meshes



http://en.wikipedia.org/wiki/File:Mesh_overview.svg

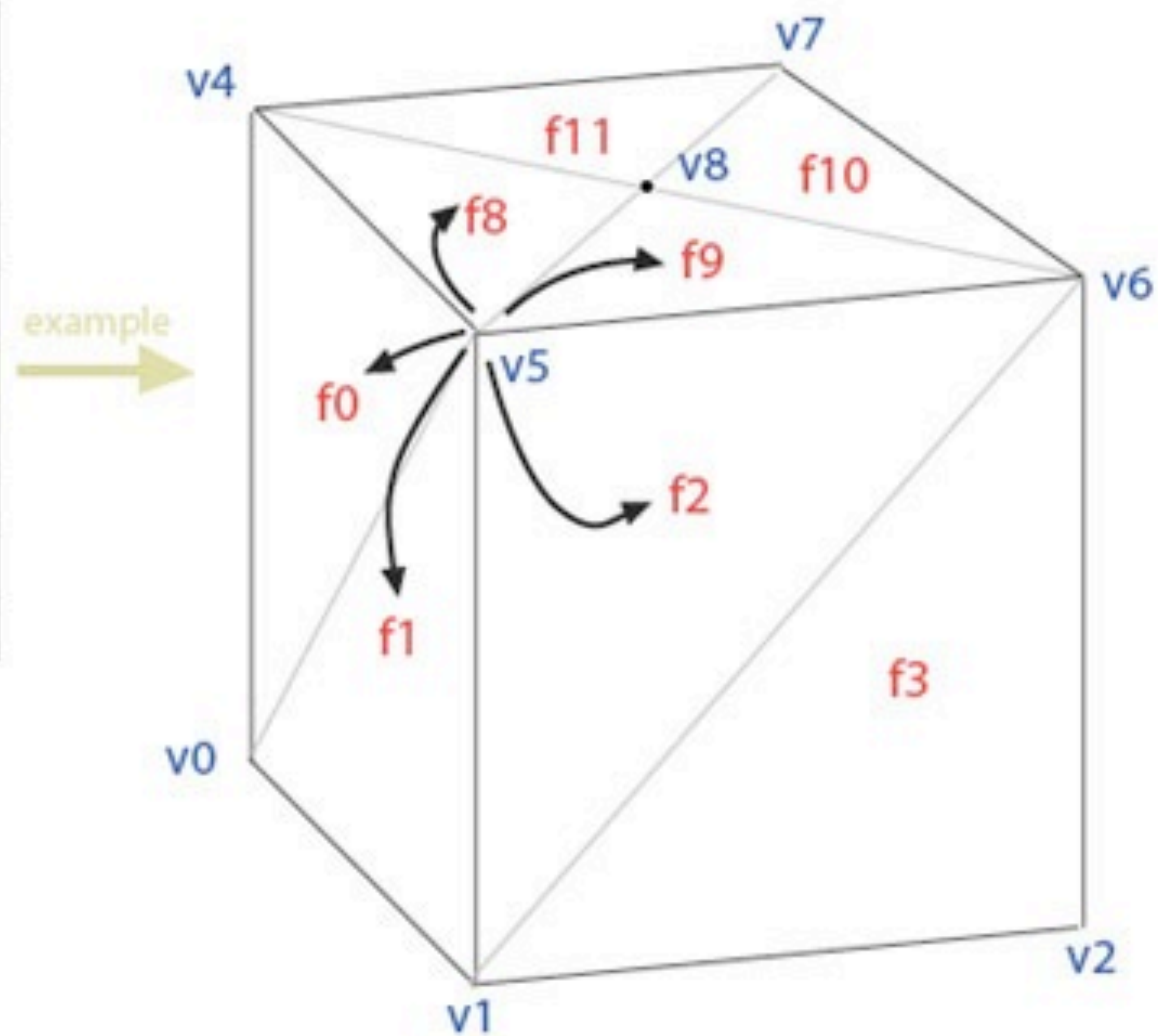
Face-Vertex Meshes

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

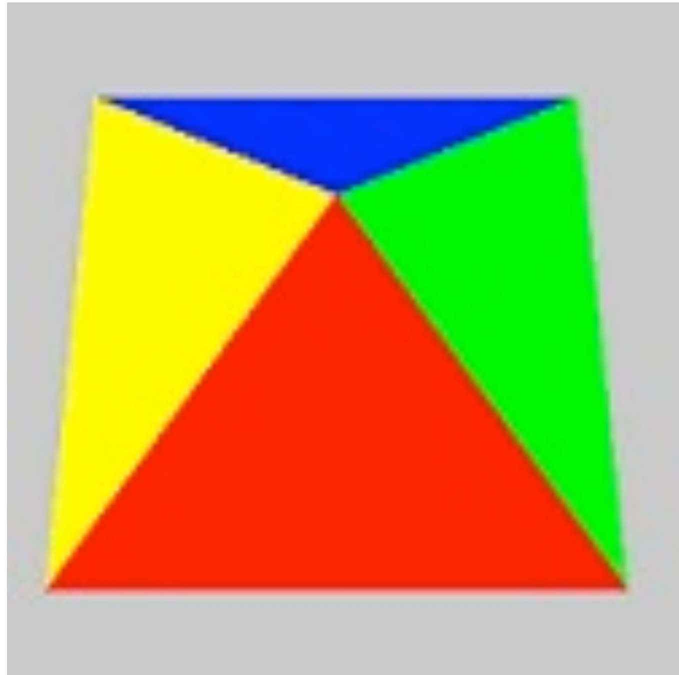
v0	0,0,0	f0 f1 f12 f15 f7
v1	1,0,0	f2 f3 f13 f12 f1
v2	1,1,0	f4 f5 f14 f13 f3
v3	0,1,0	f6 f7 f15 f14 f5
v4	0,0,1	f6 f7 f0 f8 f11
v5	1,0,1	f0 f1 f2 f9 f8
v6	1,1,1	f2 f3 f4 f10 f9
v7	0,1,1	f4 f5 f6 f11 f10
v8	.5,.5,0	f8 f9 f10 f11
v9	.5,.5,1	f12 13 14 15

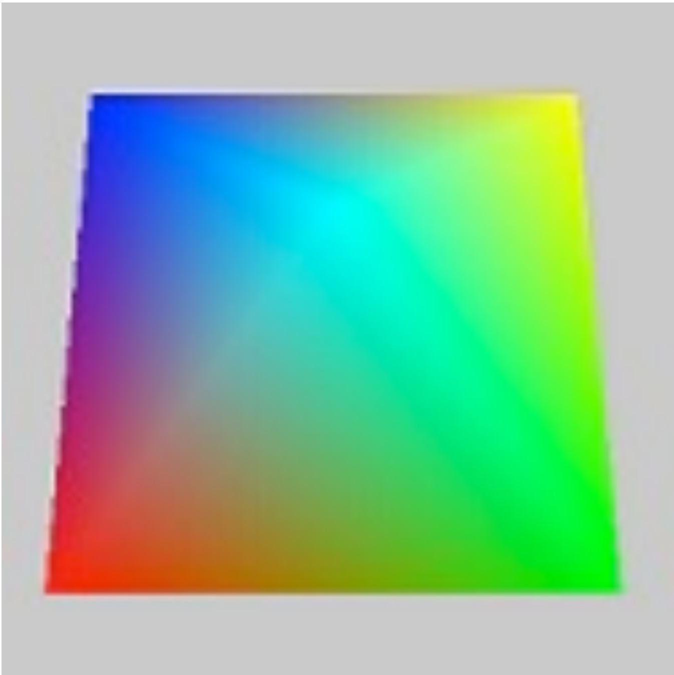


http://en.wikipedia.org/wiki/File:Mesh_fv.jpg

Polygon Meshes: optional data

- Color per vertex or per face: produces colored models
- Normal per vertex: allows free control over the normals
  - can mix smooth and sharp edges
  - wait for shading chapter ;-)
- Texture coordinates per vertex
  - wait for texture chapter ;-)

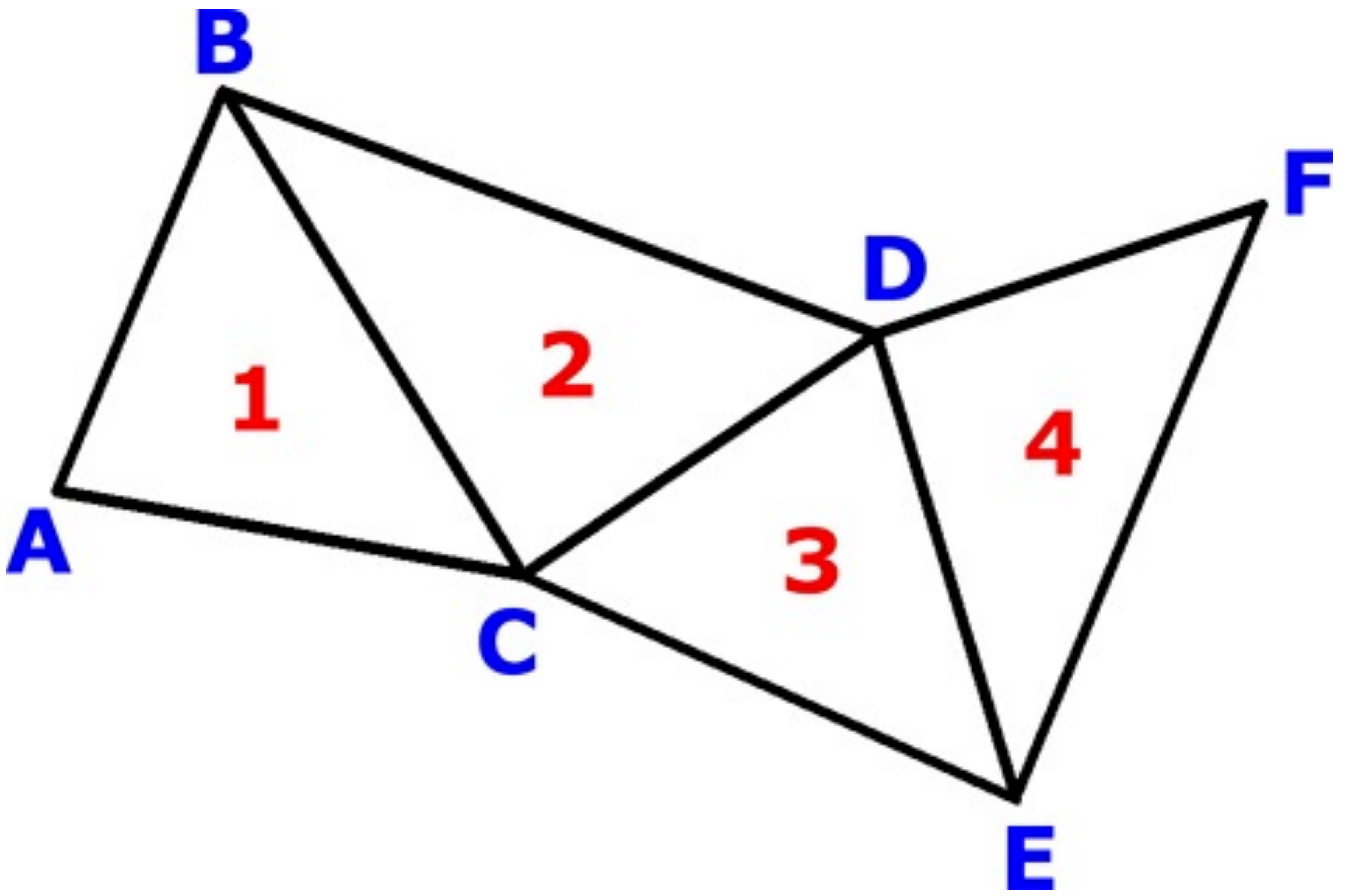




http://en.wikipedia.org/wiki/File:Triangle_Strip.png

Polygon Meshes: other descriptions

- Other representations for polygon meshes exist
  - optimized for analyzing and modyfying topology
  - optimized for accessing large models
  - optimized for fast rendering algorithms
  - optimized for graphics hardware
- Example: triangle strip
  - needs N+2 points for N polygons
  - implicit definition of the triangles
  - optimized on graphics hardware

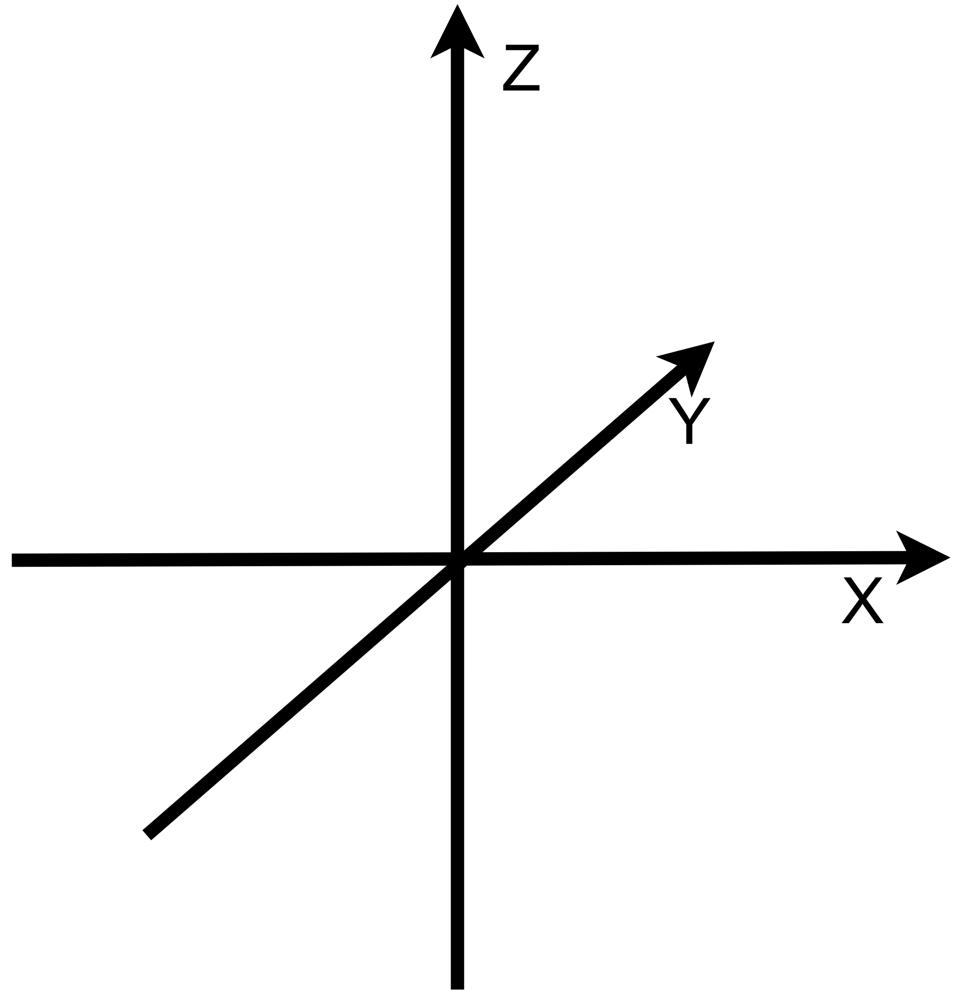


http://en.wikipedia.org/wiki/File:Triangle_Strip.png

Practical example: VRML IndexedFaceSet

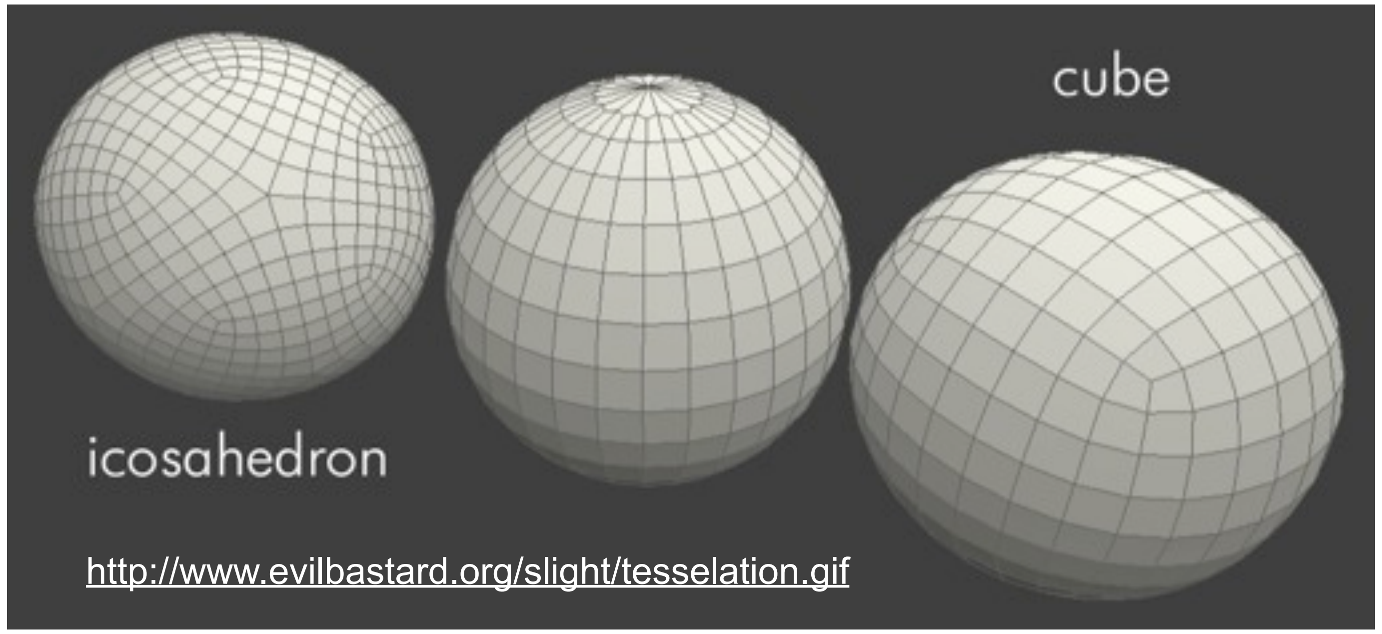
Quiz: what is given by the following piece of VRML code??

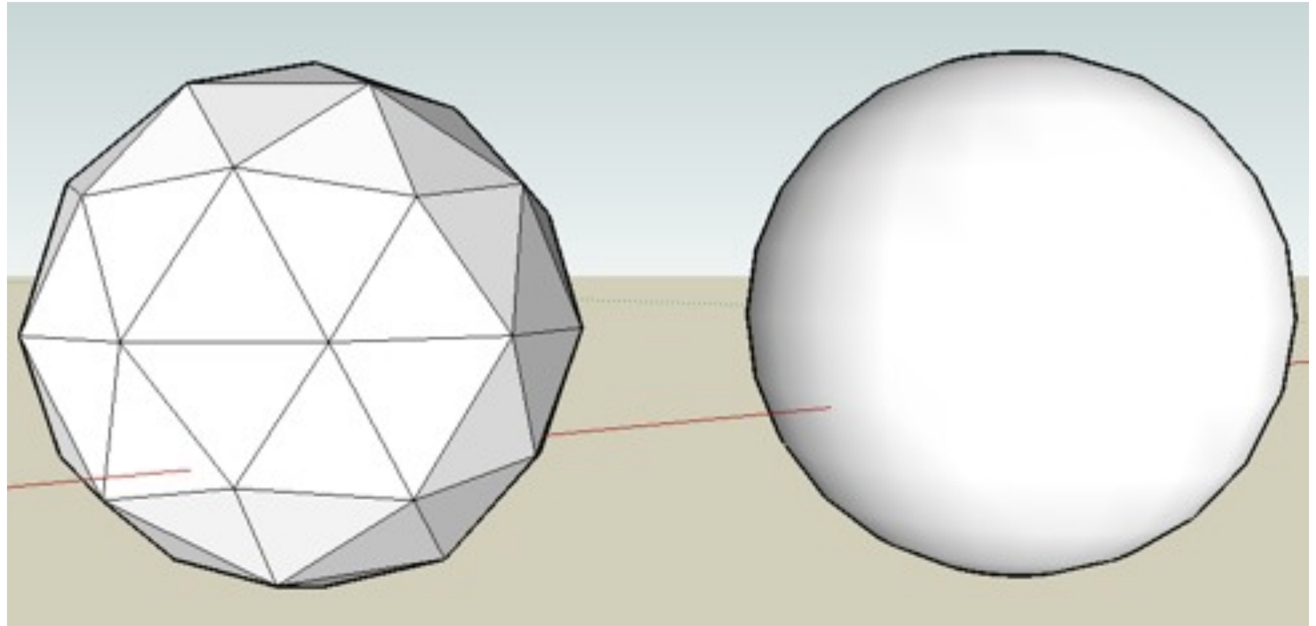
```
geometry IndexedFaceSet {
  coord Coordinate {
    point [ -1 0 1, 1 0 1, -1 0 -1, 1 0 -1, 0 1 0 ]
  }
  coordIndex [ 0, 1, 4, -1,
               1, 3, 4, -1,
               3, 2, 4, -1,
               2, 0, 4, -1,
               1, 0, 2, 3, -1 ]
}
```



Approximating Primitives by Polygon Meshes

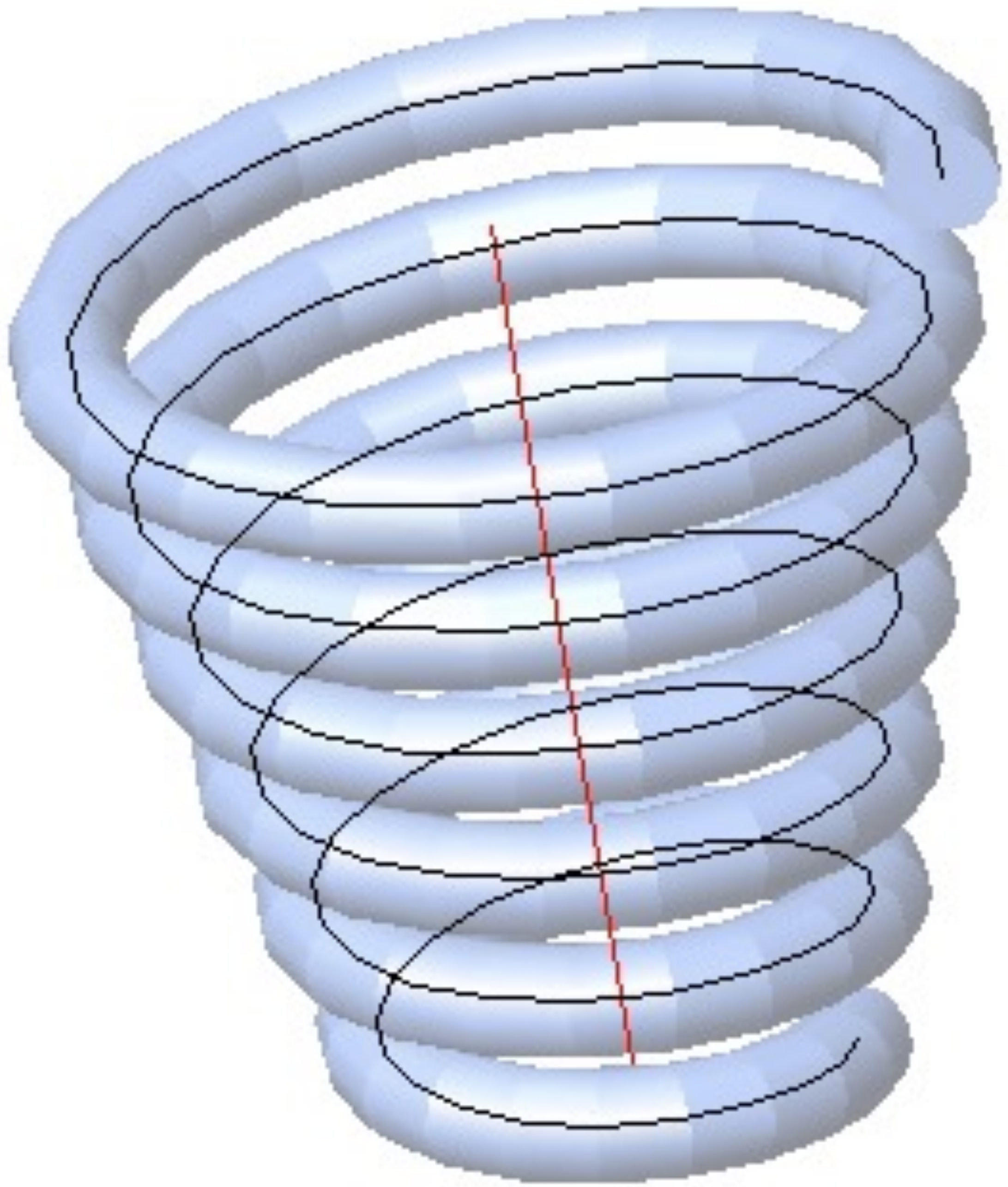
- Trivial for non-curved primitives...
- The curved surface of a cylinder, sphere etc. must be broken down into polygons somehow (Tesselation).
- Not trivial and certainly not unique!
- Goal: small polygons for strong curvature, larger ones for areas of weak curvature
  - This means ideally constant polygon size for a spere
  - Where do I know this problem from??? Hmmm...



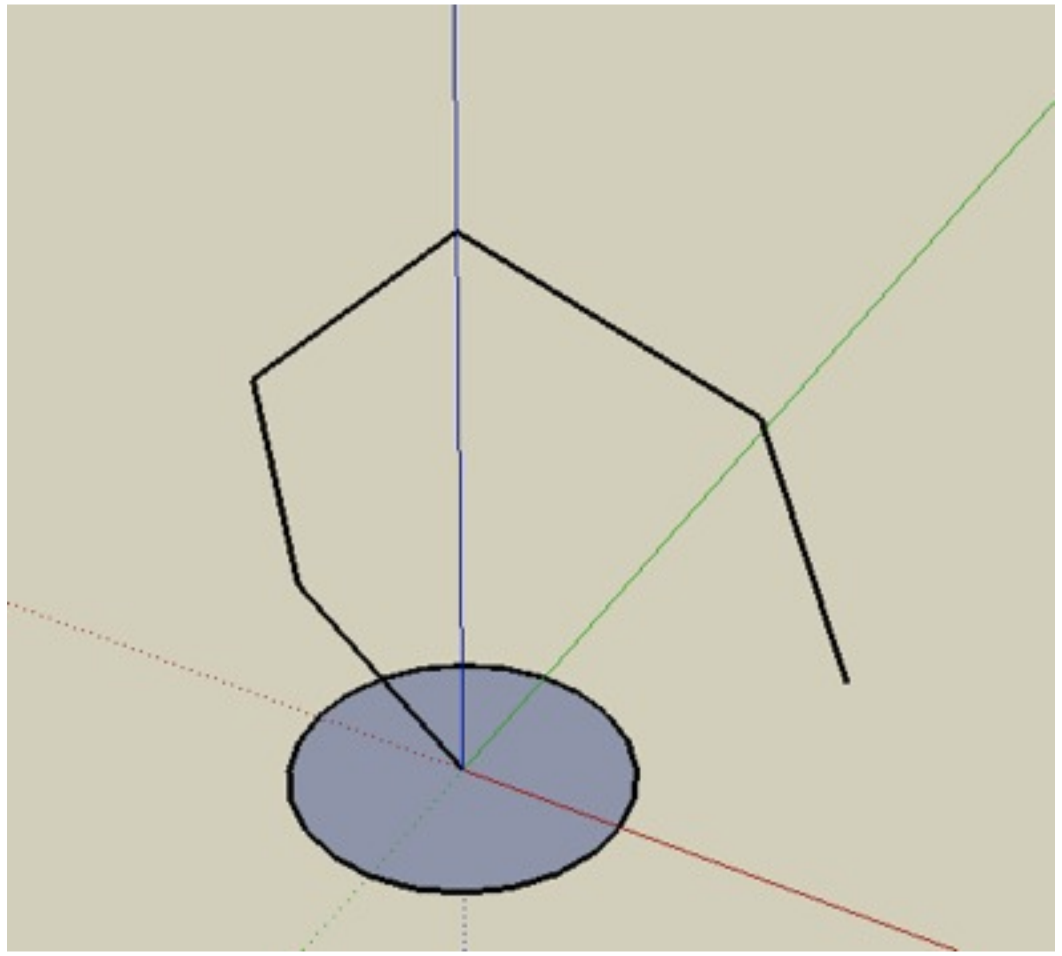


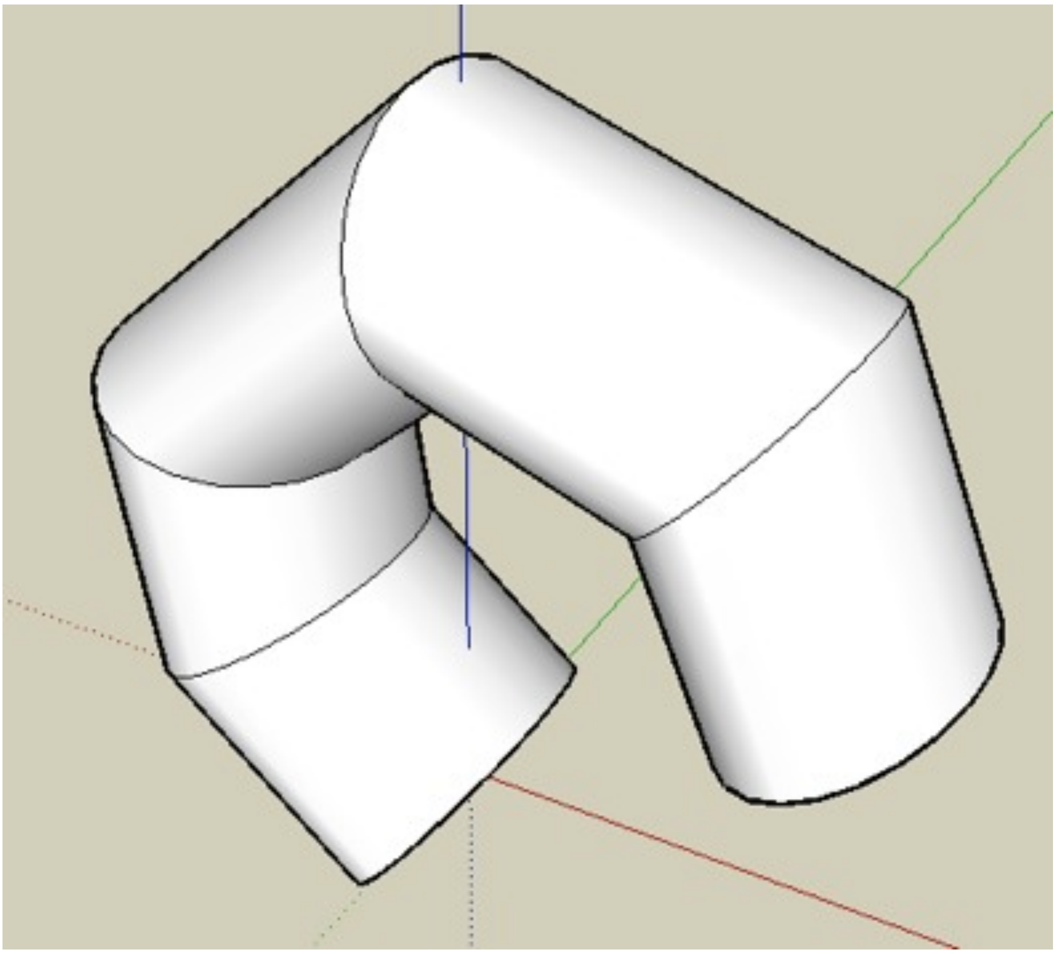
Extrusion (sweep object)

- Move a 2D shape along an arbitrary path
- possibly also scale in each step



http://www.cadimage.net/cadtutor/lisp/helix-02.gif

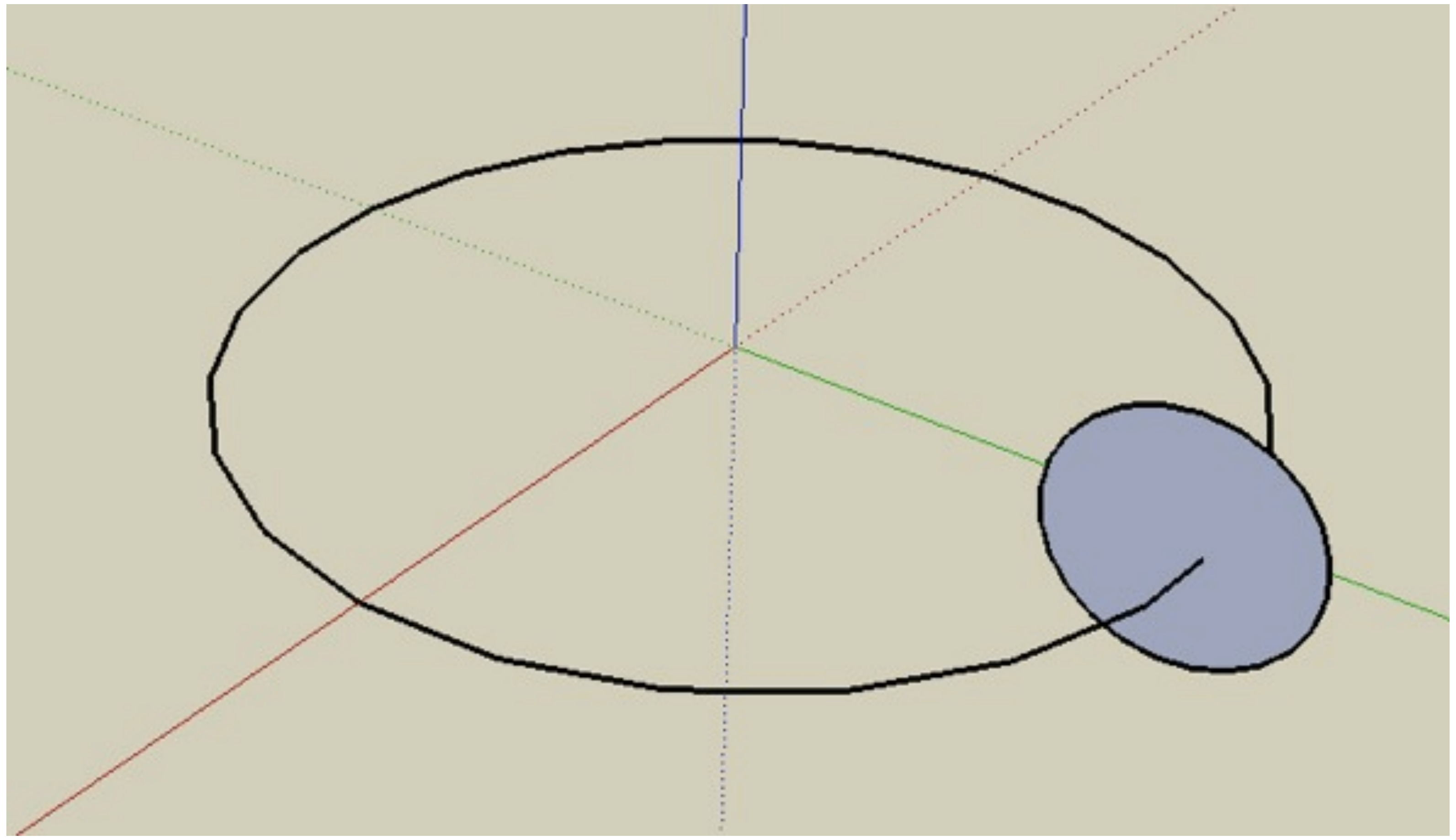


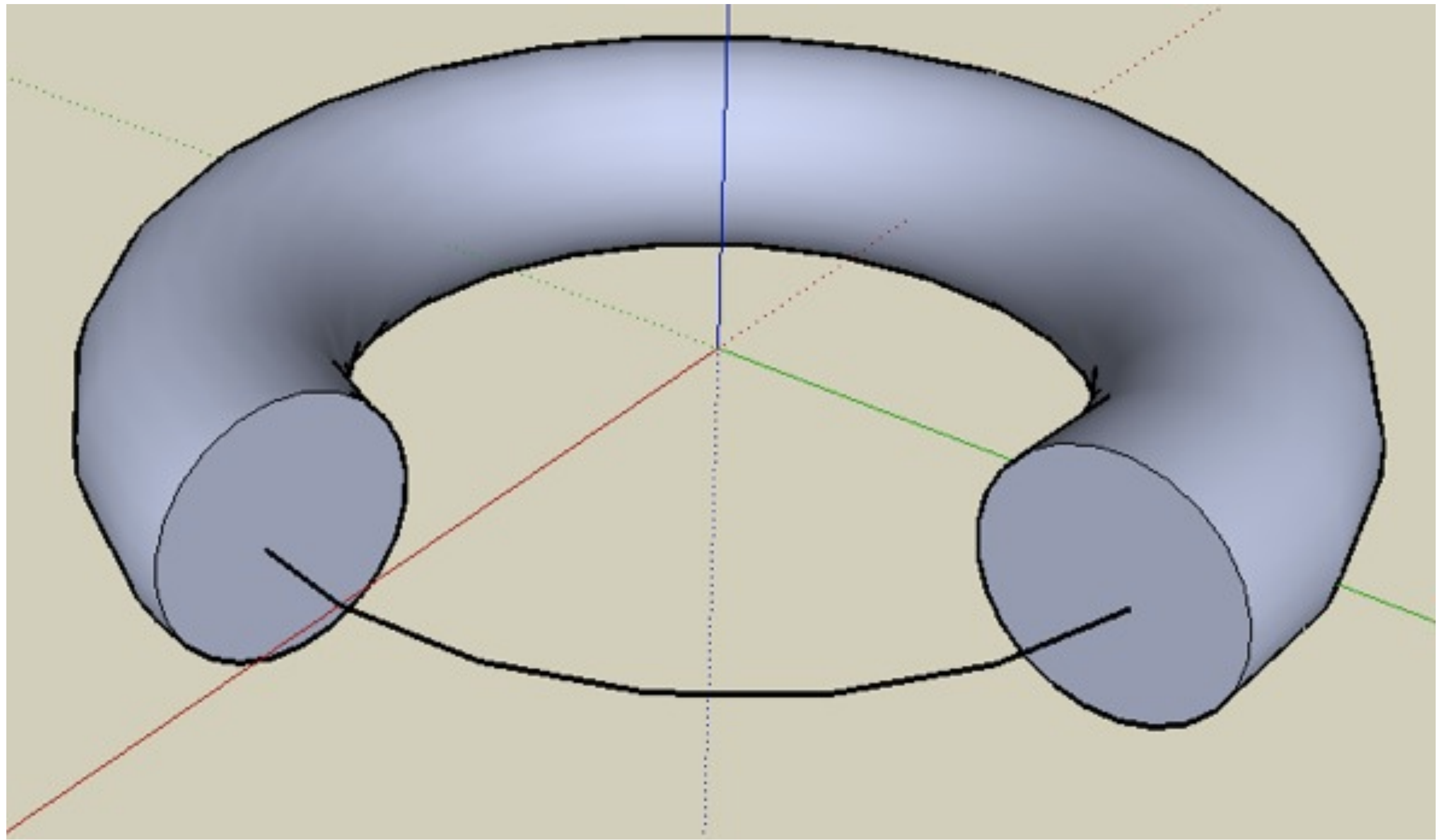


Rotation

- Rotate a 2D shape around an arbitrary axis
- Can be expressed by extrusion along a circle

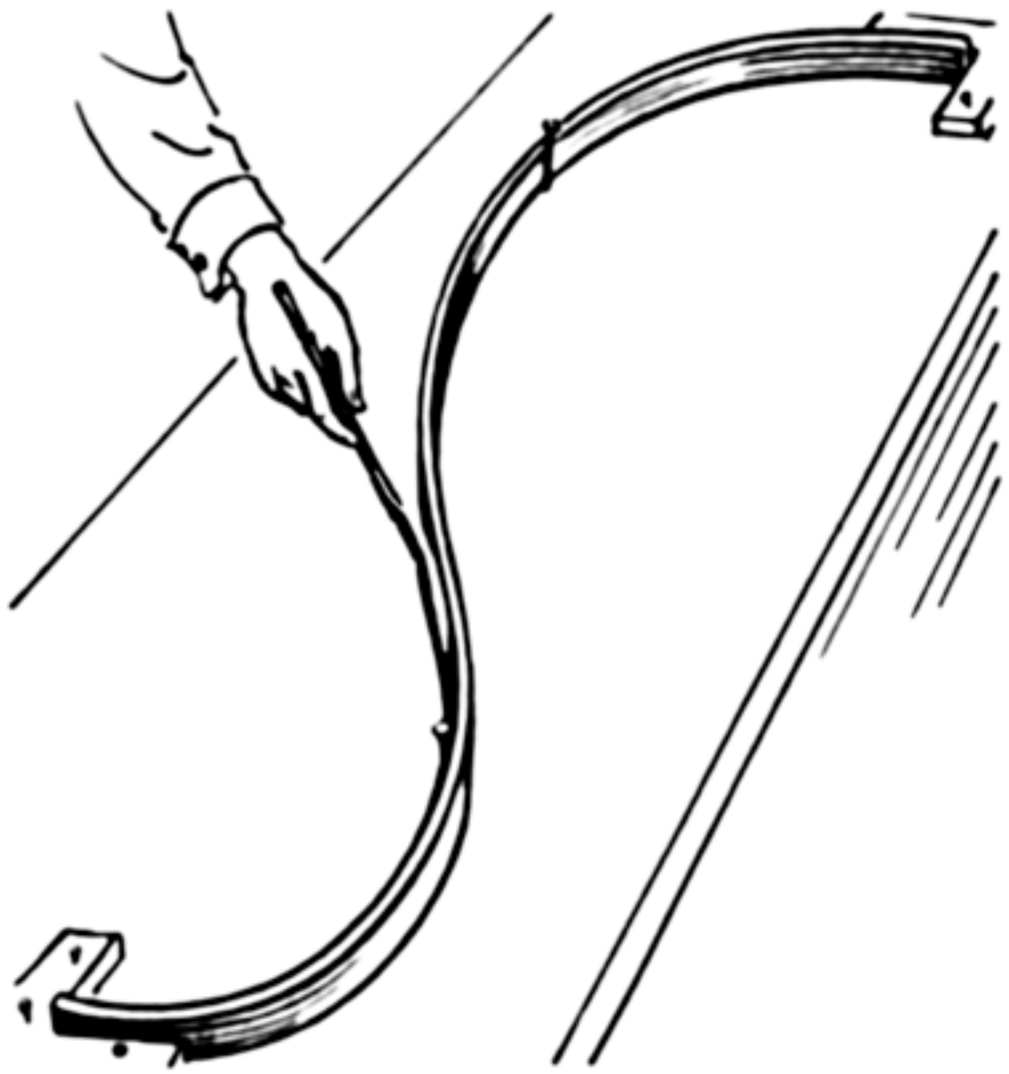
- How can we model a vase?
  –
  –
  –
- How a Coke bottle?
  –
  –
  –

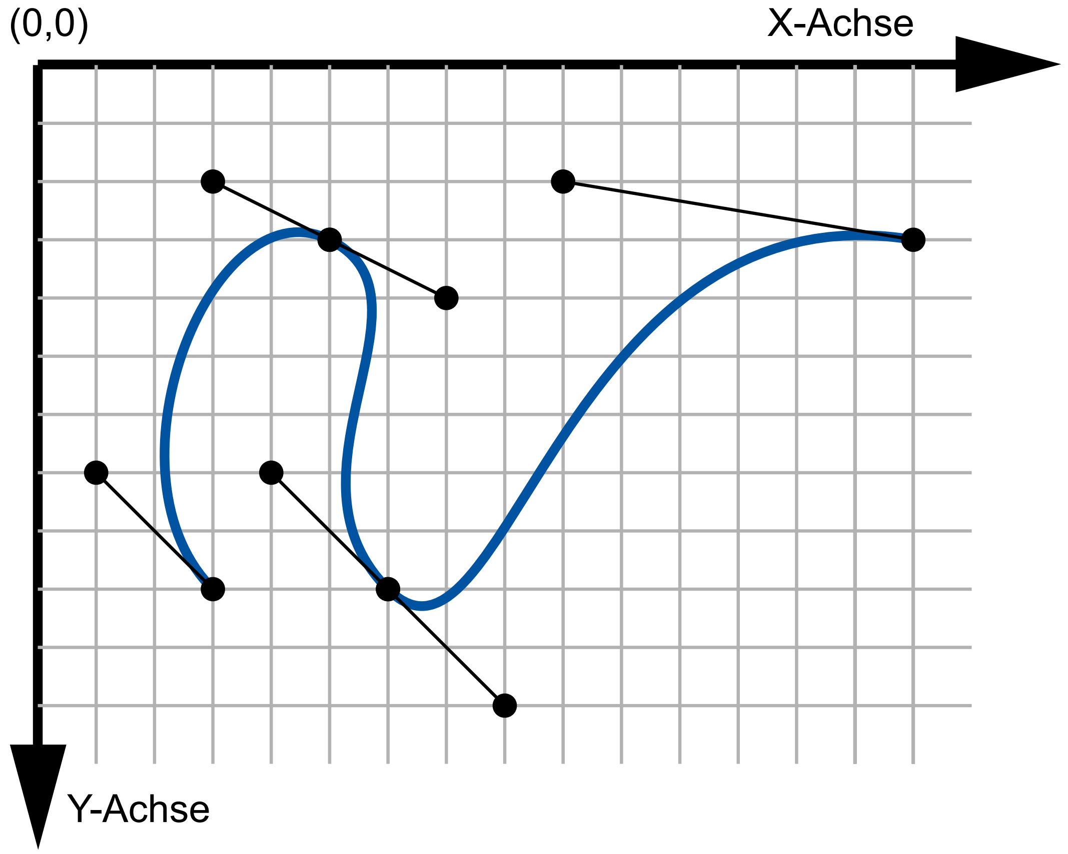




Interpolation Curves, Splines

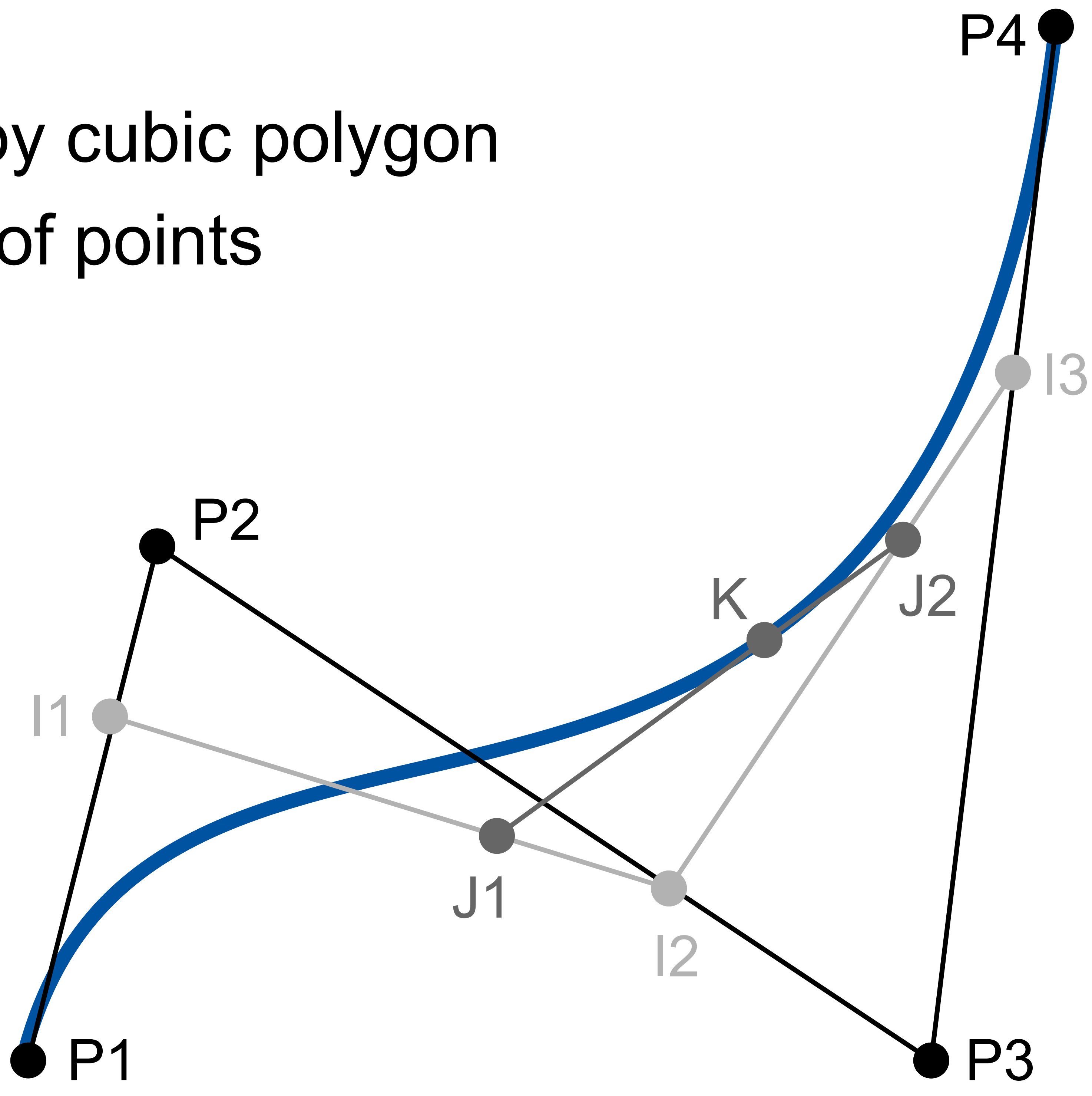
- Original idea: „Spline“ used in ship construction to build smooth shapes:
  - Elastic wooden band
  - Fixed in certain positions and directions
  - Mathematically simulated by interpolation curves
  - piecewise described by polygons
- Different types exist
- Control points my be on the line or outside of it.





Bezier Curves (and Casteljau Algorithm)

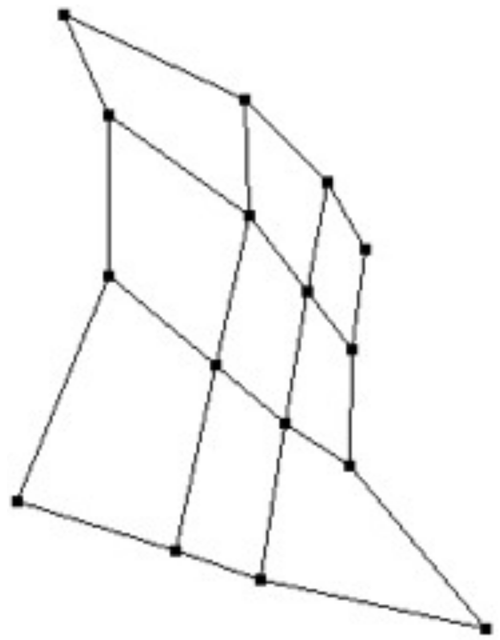
- Bezier curves first used in automobile construction (1960s, Citroen)
- Degree 1: straight line interpolated between 2 points
- Degree 2: quadratic polygon
- Degree 3: cubic bezier curve, described by cubic polygon
- Curve is always contained in convex hull of points
- Algorithm (defines line recursively):
  - I1 is linearly interpolated between P1 and P2
  - I2 ... between P2 and P3
  - I3 ... between P3 and P4
  - J1 ... between I1 and I2
  - J2 ... between I2 and I3
  - K ... between J1 and J2
  - The bezier curve is the sum of all points K
- see http://files.dmke.de/bezier.html !!!

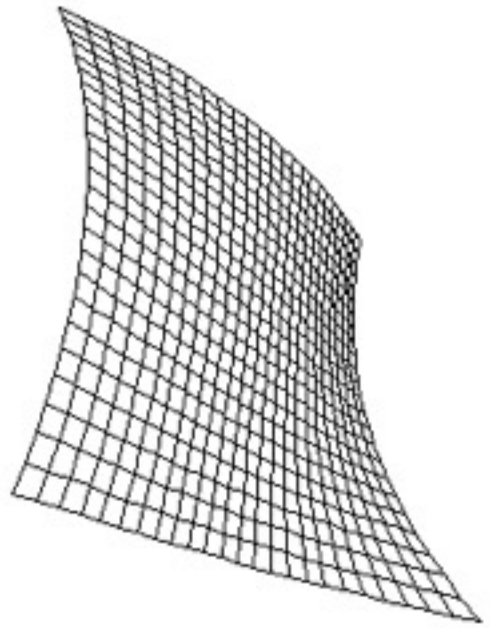


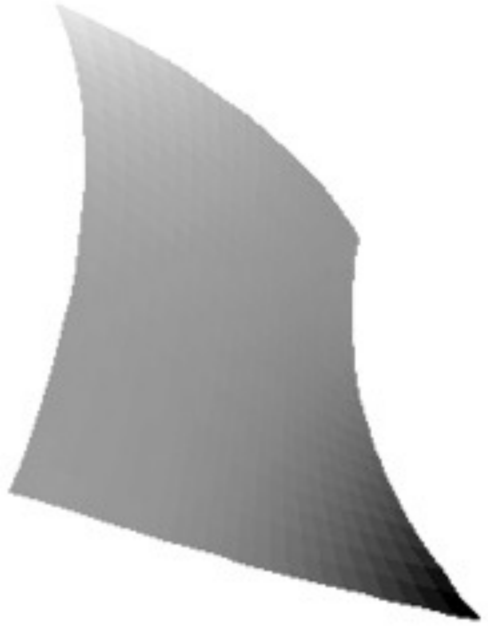
Bezier patches

- Combine 4 Bezier curves along 2 axes
- Share 16 control points
- Results in a smooth surface
- Entire surface is always contained within the convex hull of all control points
- border line is fully determined by border control points
- several patches can be combined
  - connect perfectly if border control points are the same.

- Other interpolation surfaces based on other curves
- advantage: move just one control point to deform a larger surface...

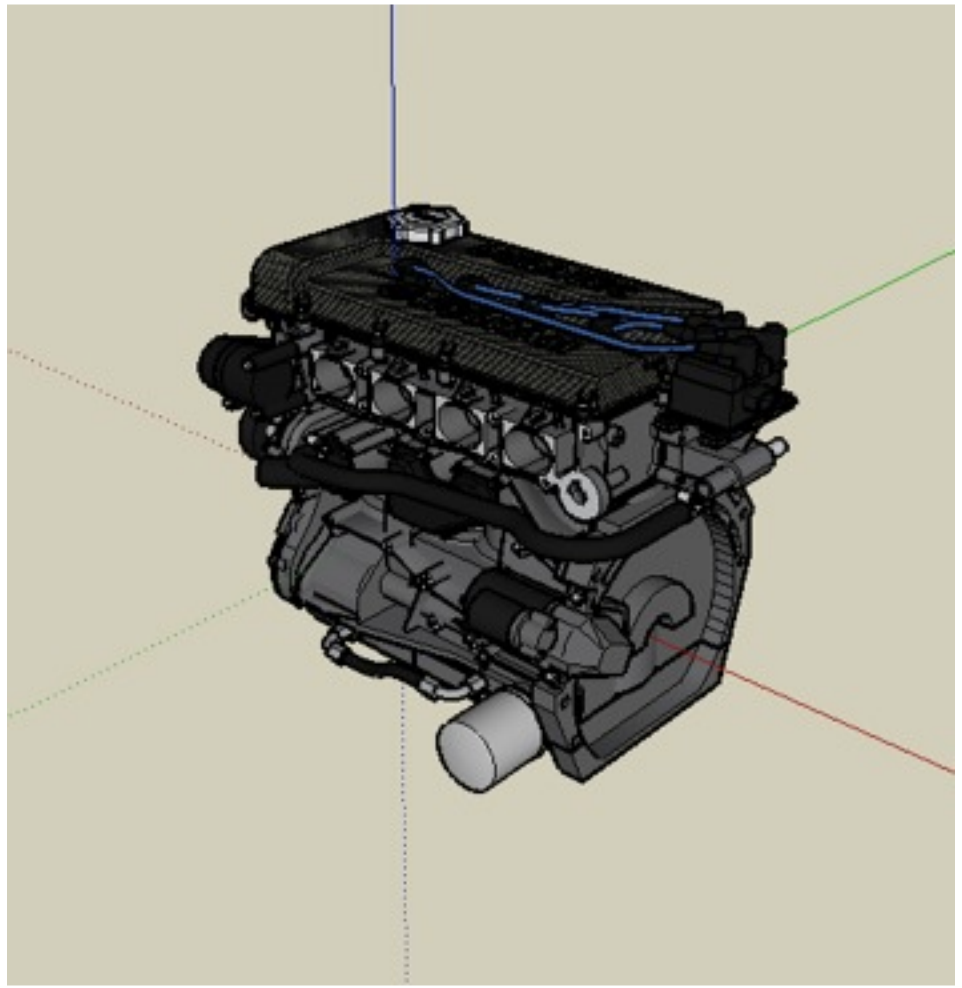


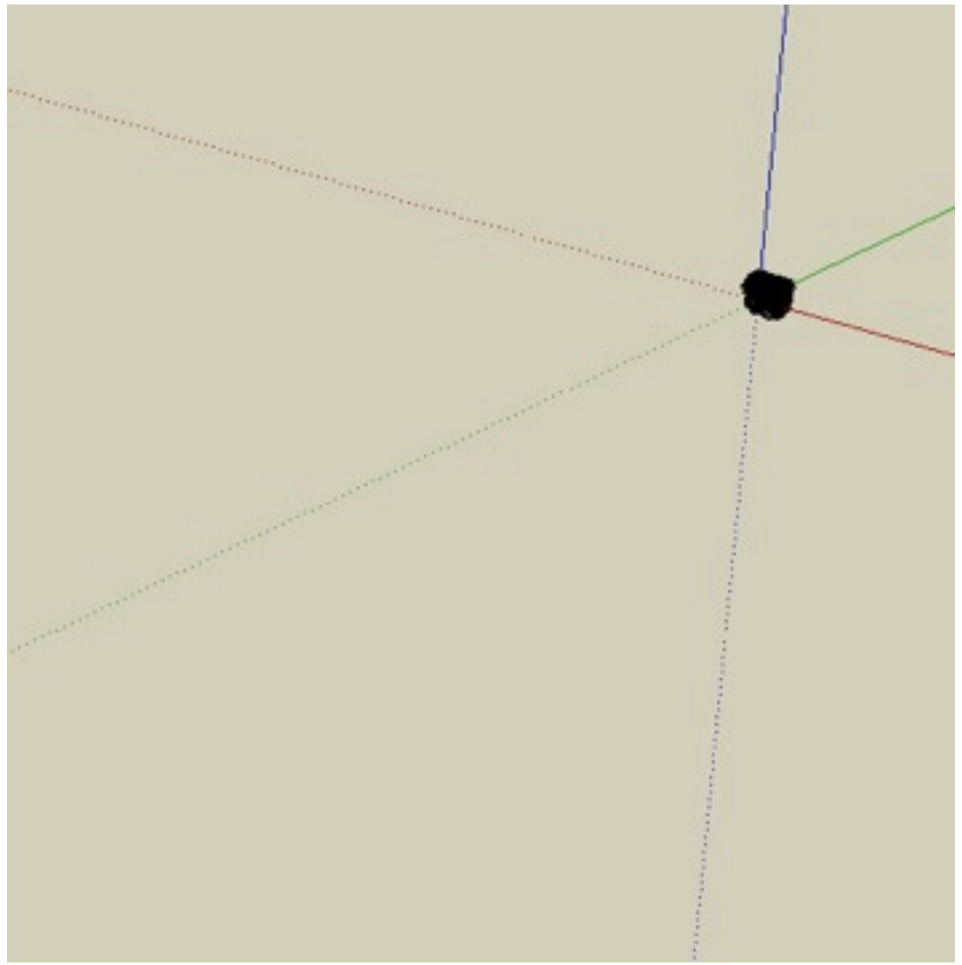




Levels of Detail

- Assume you have a very detailed model
- from close distance, you need all polygons
- from a far distance, it only fills a few pixels
- How can we avoid drawing all polygons?
  –
  –
  –
  –



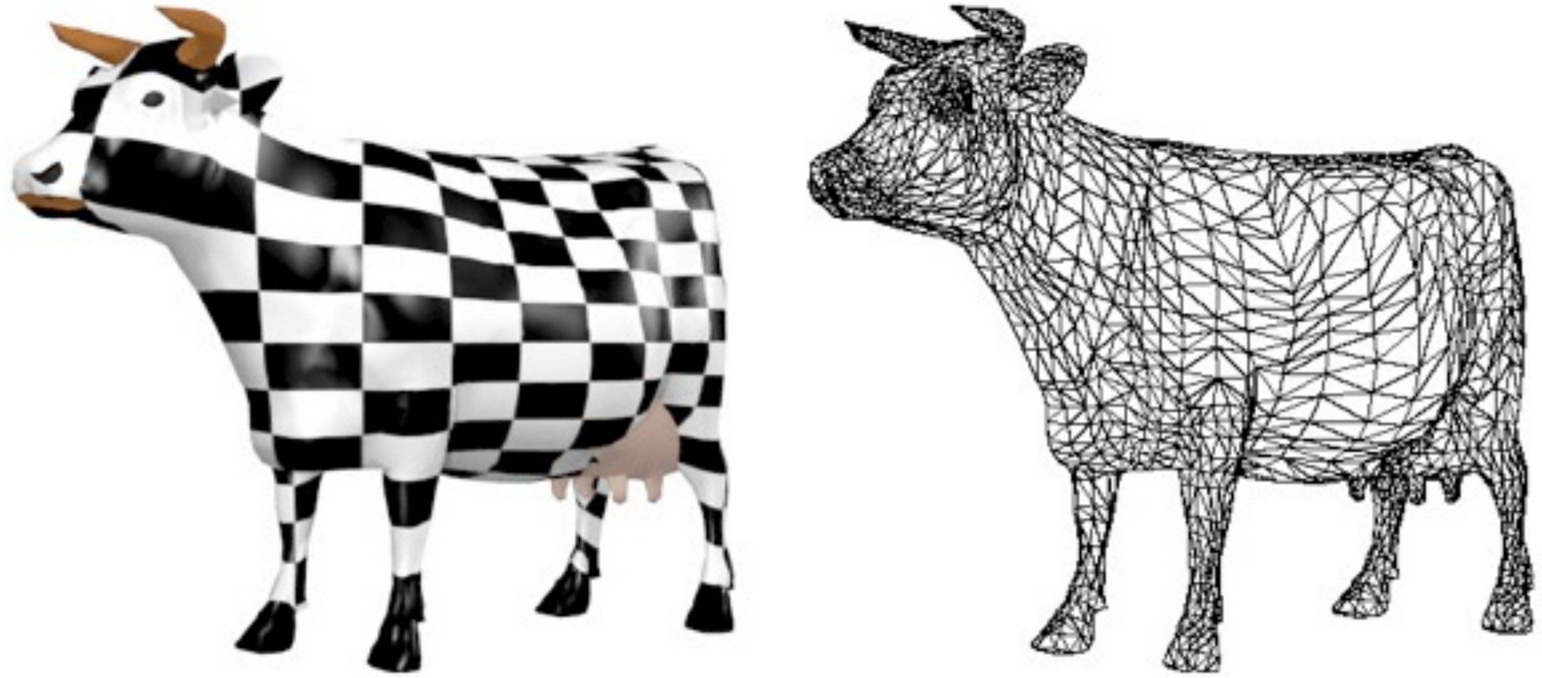


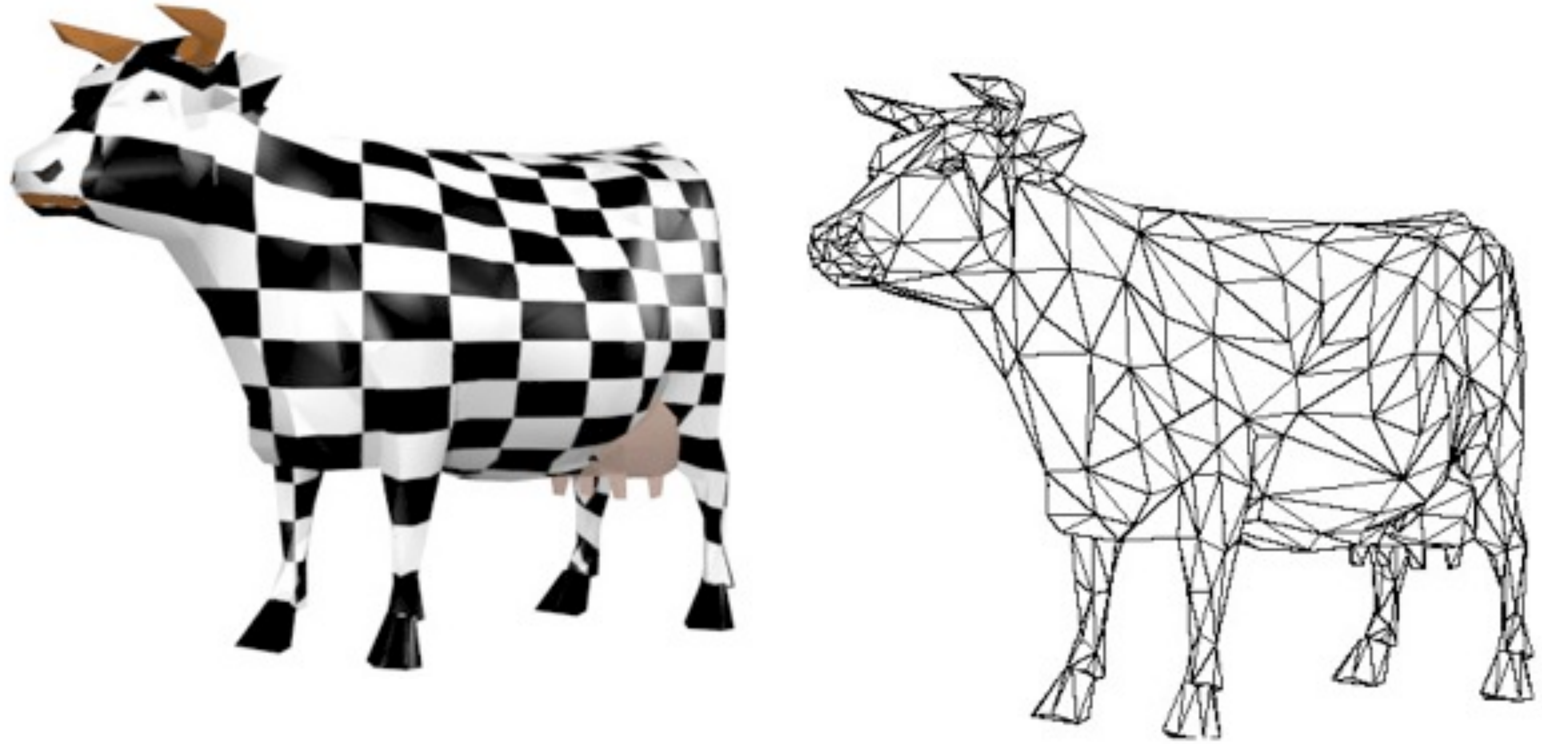
Mesh reduction

- Original: ~5.000 polygons
- Reduced model: ~1.000 polygons
- ==> about 80% reduction

- Very strong reductions possible, depending on initial mesh

- Loss of shape if overdone



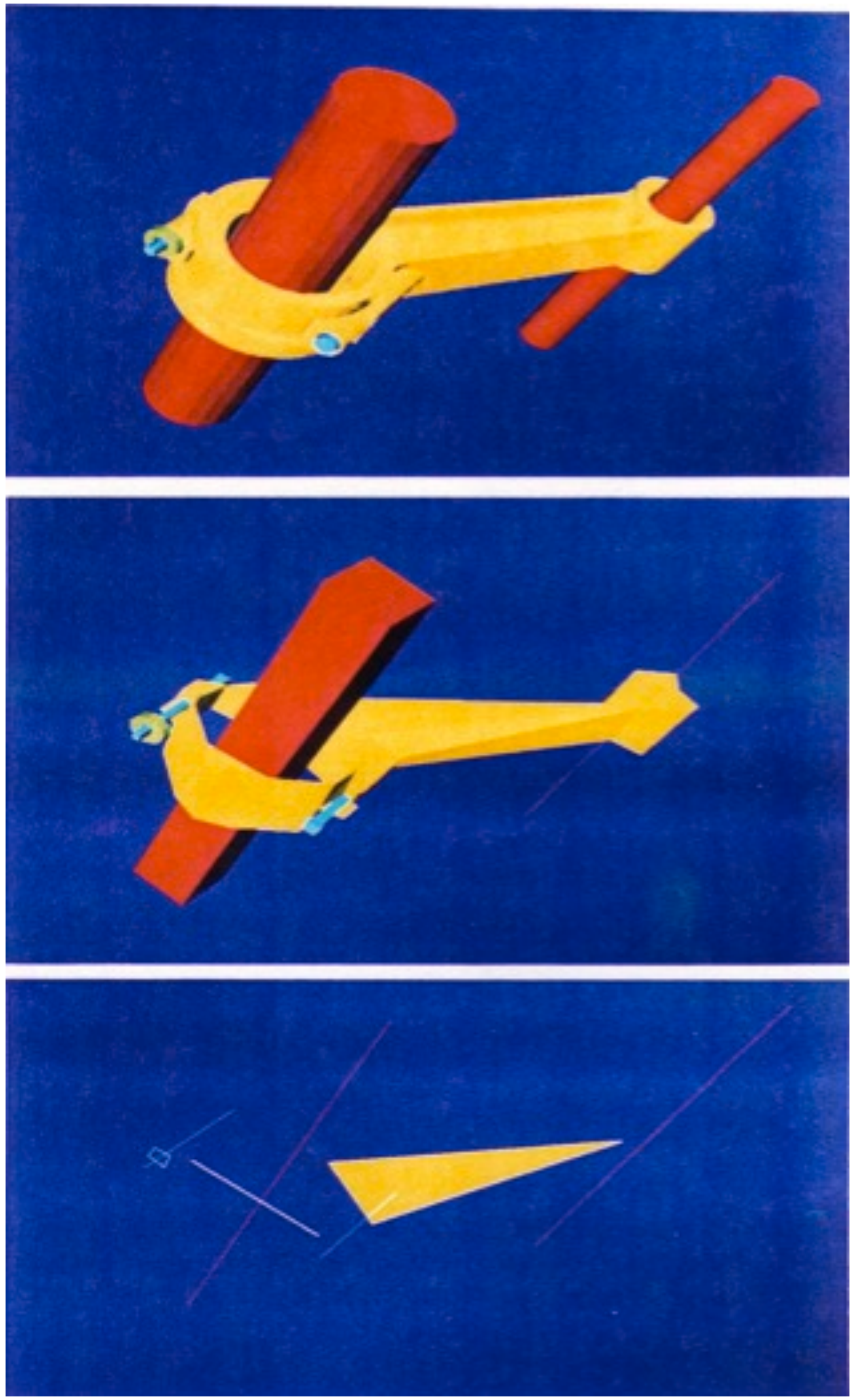


http://www.okino.com/conv/polygon_reduction/geoman2/polygon_reduction_tutorial1.htm

A method for polygon reduction

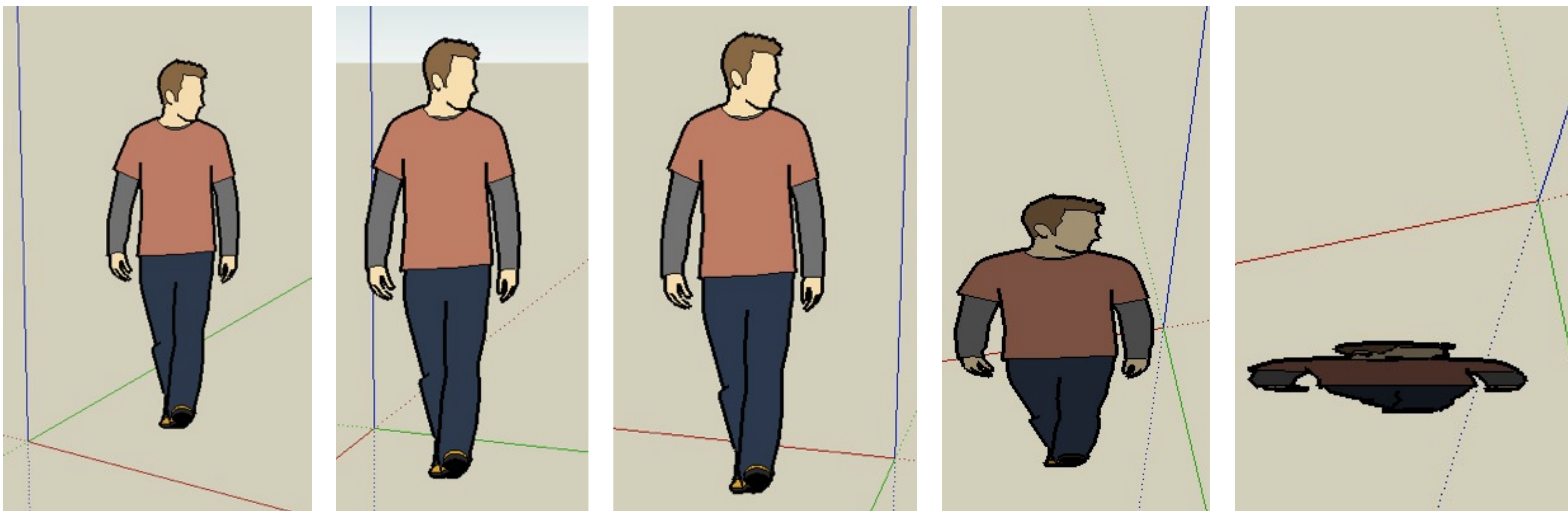
- Rossignac and Borell, 1992, „Vertex clustering“
- subdivide space into a regular 3D grid
- for each grid cell, melt all vertices into one
  - choose center of gravity of all vertices as new one
  - triangles within one cell disappear
  - triangles across 2 cells become edges (i.e. disappear)
  - triangles across 3 cells remain
- good guess for the minimum size of a triangle
  - edge length roughly = cell size
- yields constant vertex density ion space
- does not pay attention to curvature

- more: http://mkrus.free.fr/CG/LODS/xrds/

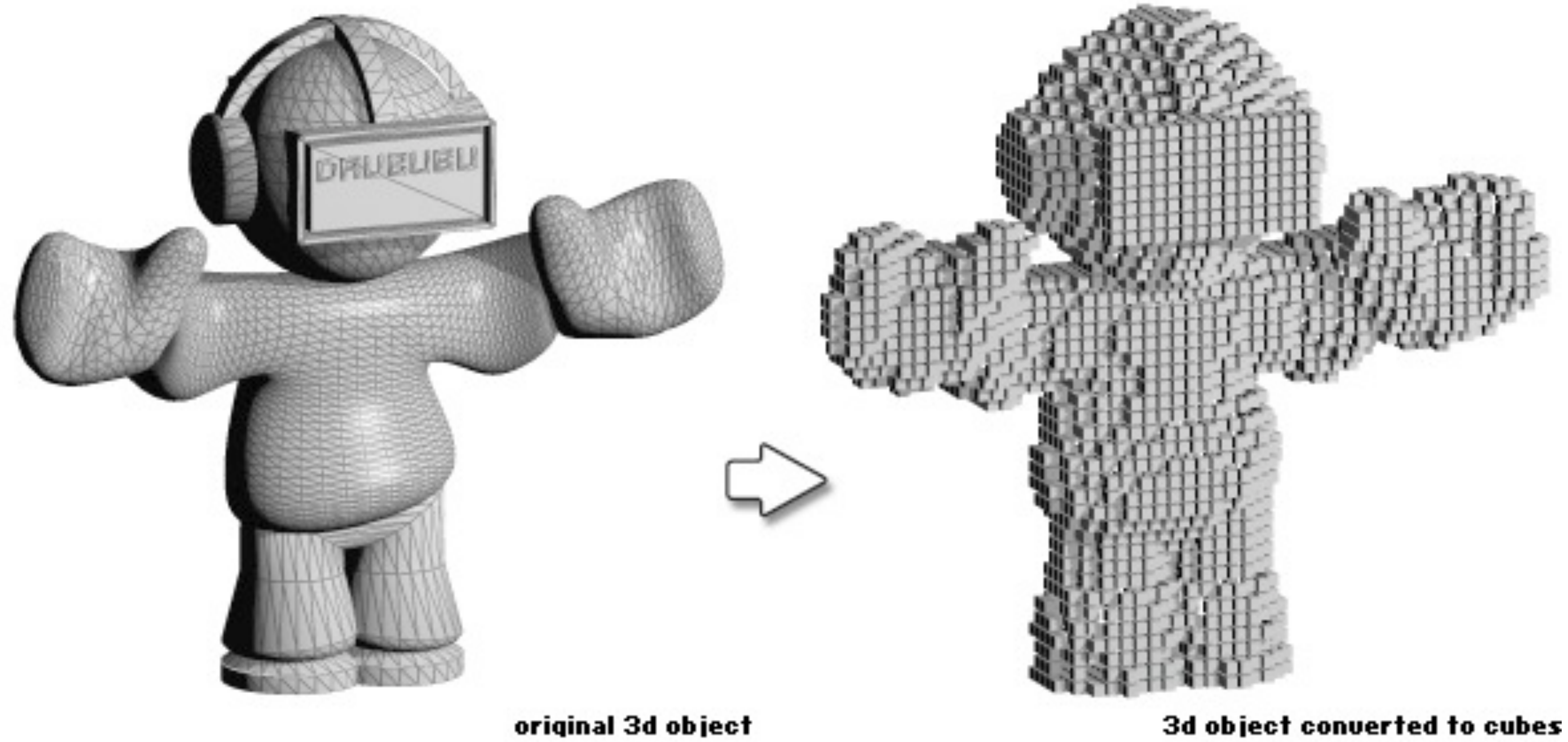


Billboard

- A flat object which is always facing you
- Very cheap in terms of polygons (2 triangles)
- Needs a meaningful texture
- Example (from SketchUp): guy in the initial empty world rotates about his vertical axis to always face you



Voxel data

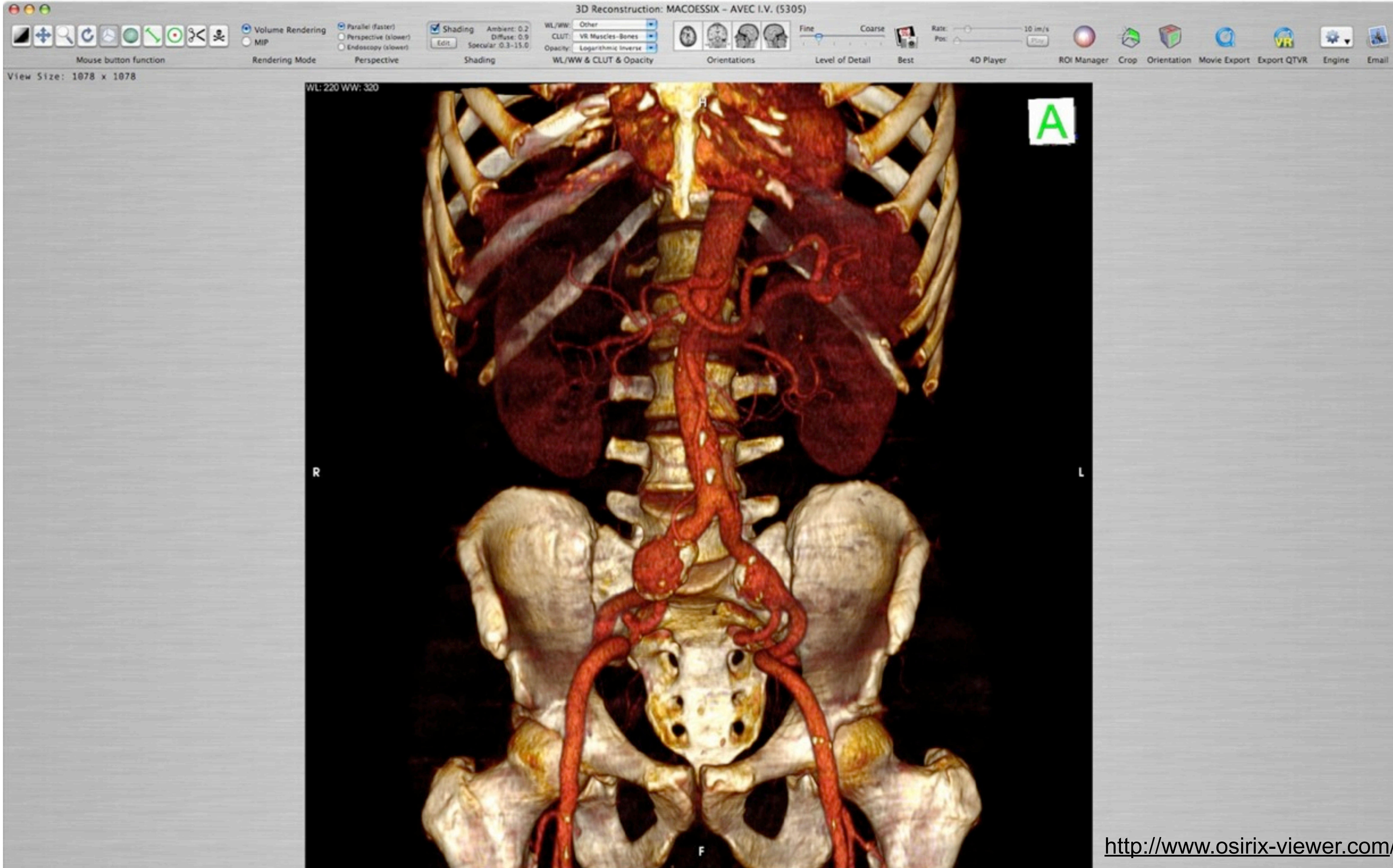


- „Voxel“ = „Volume“ + „Pixel“, i.e., voxel = smallest unit of volume
- Regular 3D grid in space
- Each cell is either filled or not
- Memory increases (cubic) with precision

- Easily derived from CSG models
- Also the result of medical scanning devices
  – MRI, CT, 3D ultrasonic

- Volume rendering = own field of research
- Surface reconstruction from voxels

http://www.drububu.com/tutorial/voxels.html



http://www.osirix-viewer.com/

Point-based graphics

- Objects represented by point samples of their surface („Surfels“)
- Each point has a position and a color
- Surface can be visually reconstructed from these points
  - purely image-based rendering
  - no mesh structure
  - very simple source data (x,y,z,color)

- Point-data is acquired e.g., by 3D cameras
- Own rendering techniques
- Own pipeline
- ==> own lecture ;-)

http://www.crs4.it/vic/data/images/img-exported/stmatthew_4px_full_shaded2.png