15-294 Rapid Prototyping Technologies:

STL Files and Slicing Software

Dave Touretzky
Computer Science
Carnegie Mellon University

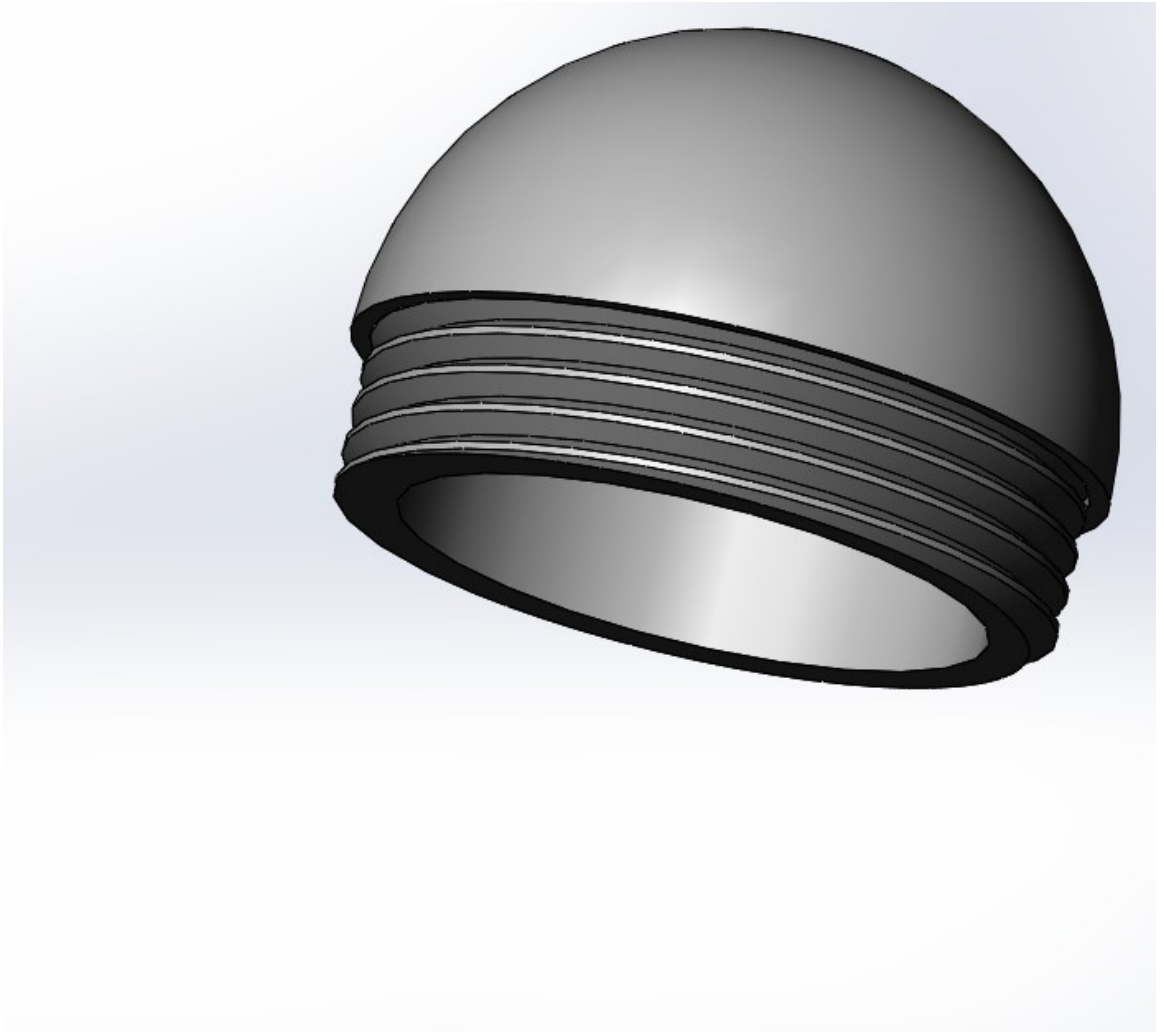
The STL File Format

- StereoLithography file

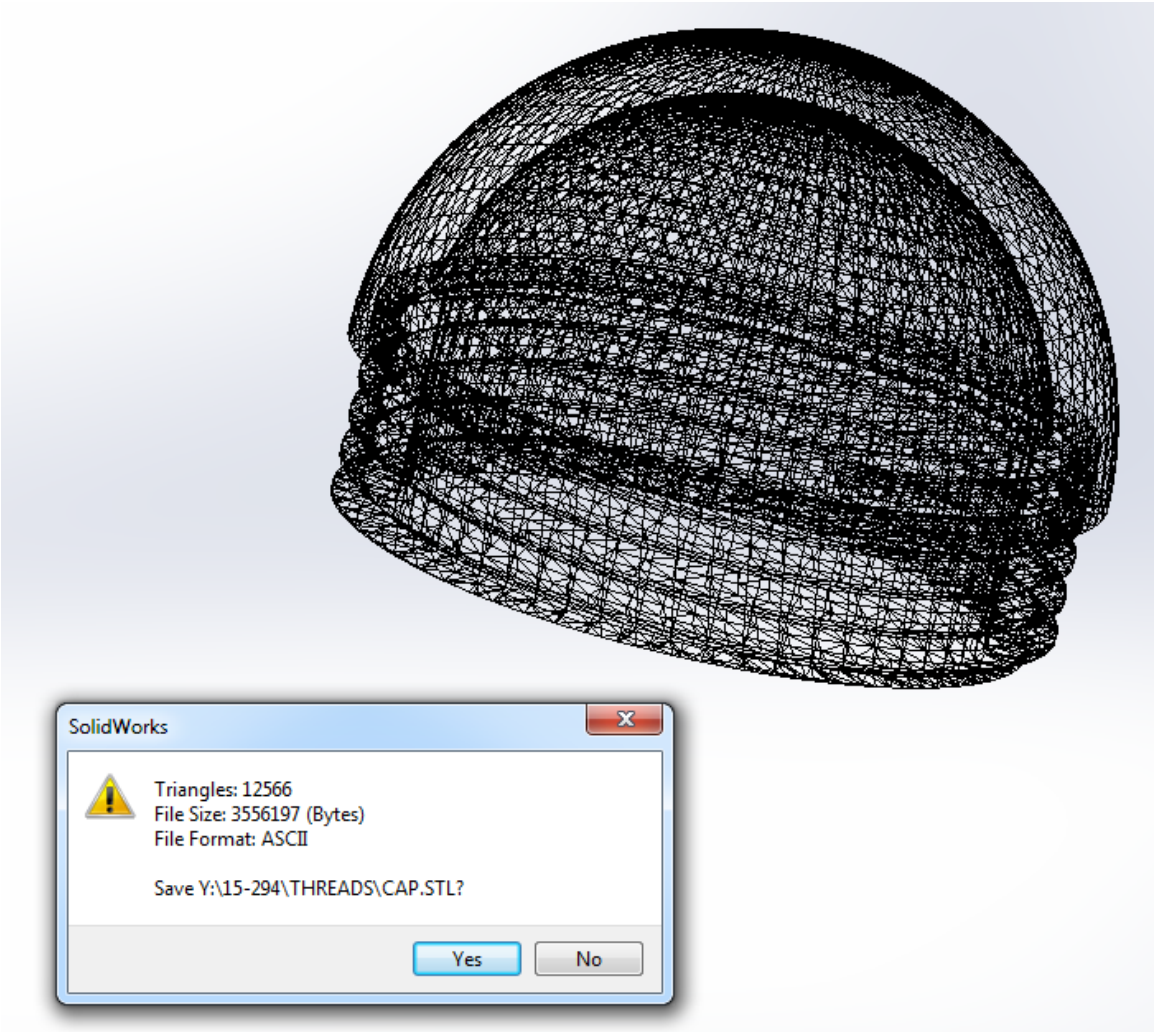
  – or –

  Standard Tesselation Language
- Originally developed by 3D Systems.
- Now widely used for describing 3D surfaces for CAD or printing.
- Two flavors: ASCII or Binary.

Object With Complex Surfaces



Triangular Tesselation from SolidWorks “Save As STL” Dialog



ASCII STL File

solid <*name*>
facet normal n_i n_j n_k
  outer loop
    vertex $v1_x$ $v1_y$ $v1_z$
    vertex $v2_x$ $v2_y$ $v2_z$
    vertex $v3_x$ $v3_y$ $v3_z$
  endloop
endfacet
…
endsolid <*name*>

Binary STL File

UINT8[80] – Header (must not begin with “solid”)

UINT32 – Number of triangles

for each triangle:

  REAL32[3] – Normal vector

  REAL32[3] – Vertex 1 x,y,z

  REAL32[3] – Vertex 2 x,y,z

  REAL32[3] – Vertex 3 x,y,z

  UINT16 – Attribute byte count (typically zero)

Some variants of STL store color information in the attribute byte count.

Python Code to Write STL Files

- See demo files in class STL directory.
- Rules for STL creation:
  - Triangles are flat (planar). To make a curved surface, use more triangles.
  - Every vertex belongs to at least two triangles.
  - No vertex can touch an edge of another triangle.

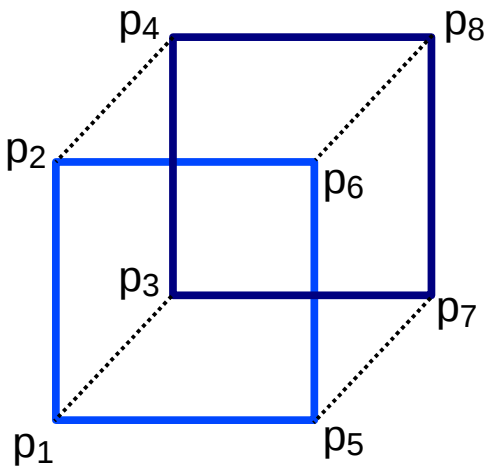
Example: Making a Cube

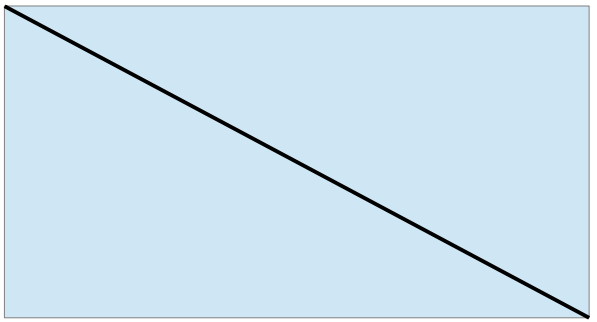
```
s = 3.0    # length of a side

# Eight corner points of a cube

p1 = (0, 0, 0)
p2 = (0, 0, s)
p3 = (0, s, 0)
p4 = (0, s, s)
p5 = (s, 0, 0)
p6 = (s, 0, s)
p7 = (s, s, 0)
p8 = (s, s, s)
```

Cube Faces

Six faces of a cube; each face yields two triangles.

```
[
    [p1, p3, p7, p5],
    [p1, p5, p6, p2],
    [p5, p7, p8, p6],
    [p7, p3, p4, p8],
    [p1, p2, p4, p3],
    [p2, p6, p8, p4],
 ]
```



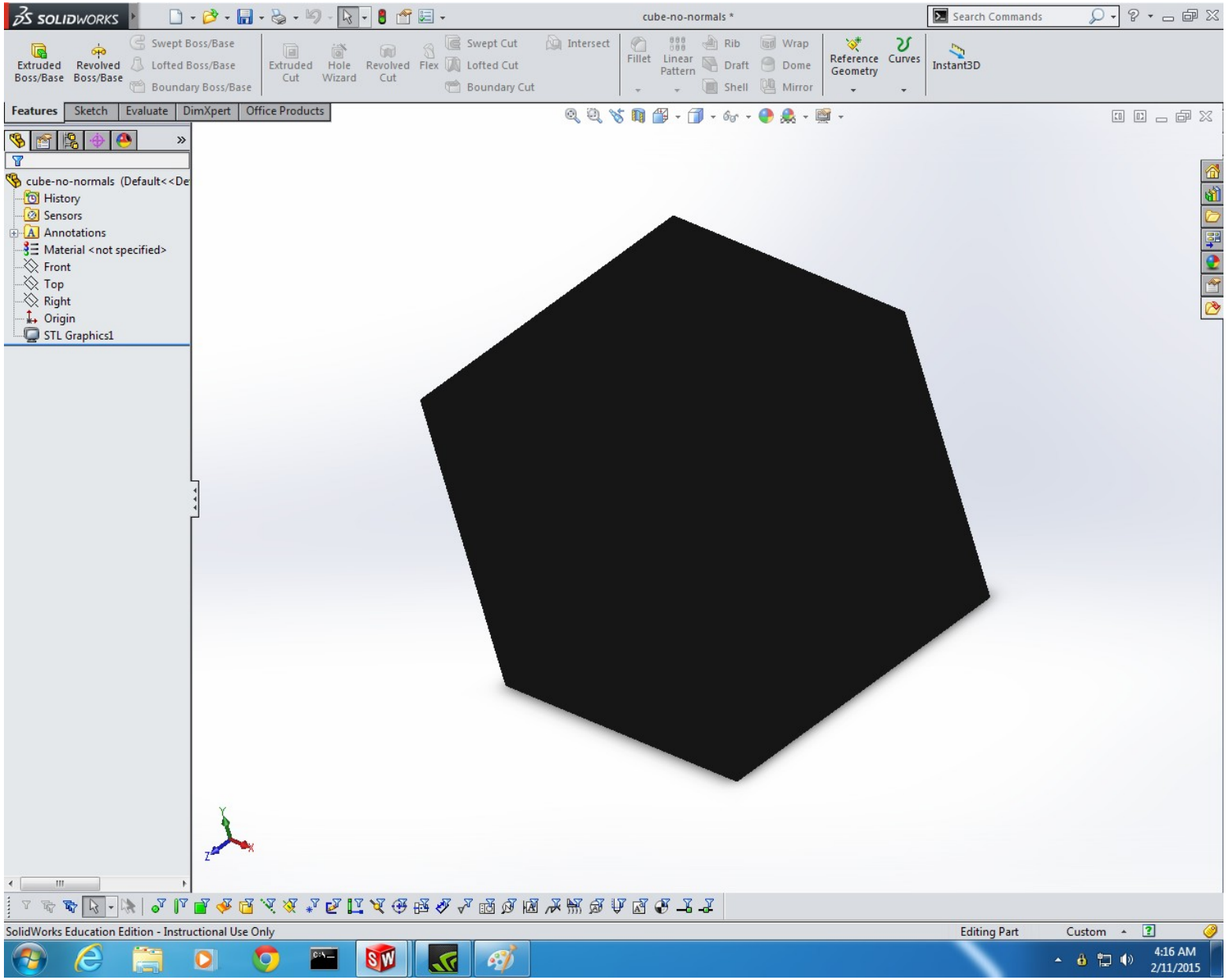


Dividing a rectangle into two triangles.

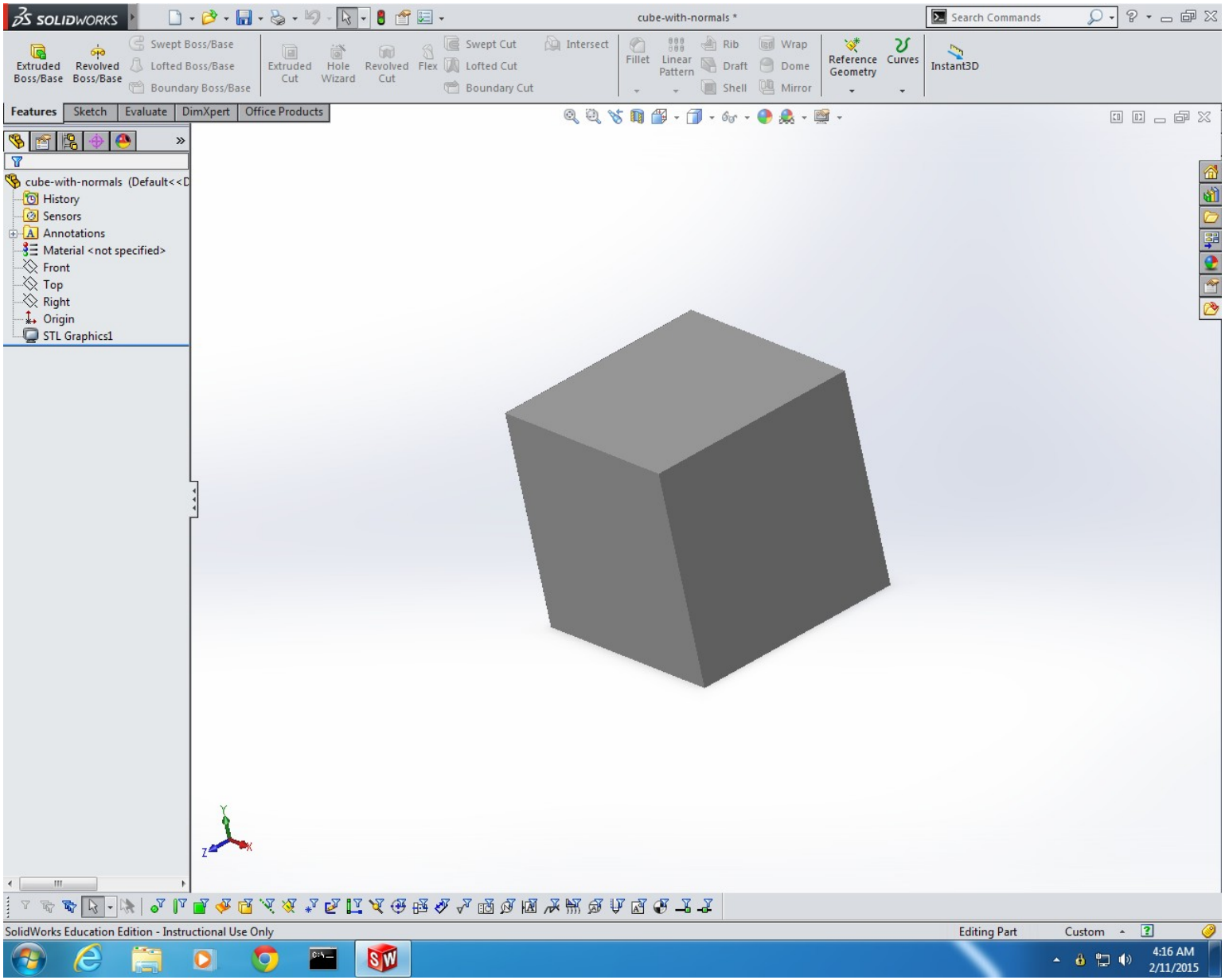
Writing the STL File: cube_demo.py

```
with open('cube.stl', 'wb') as fp:
    writer = ASCII_STL_Writer(fp)
    writer.add_faces(get_cube())
    writer.close()
```

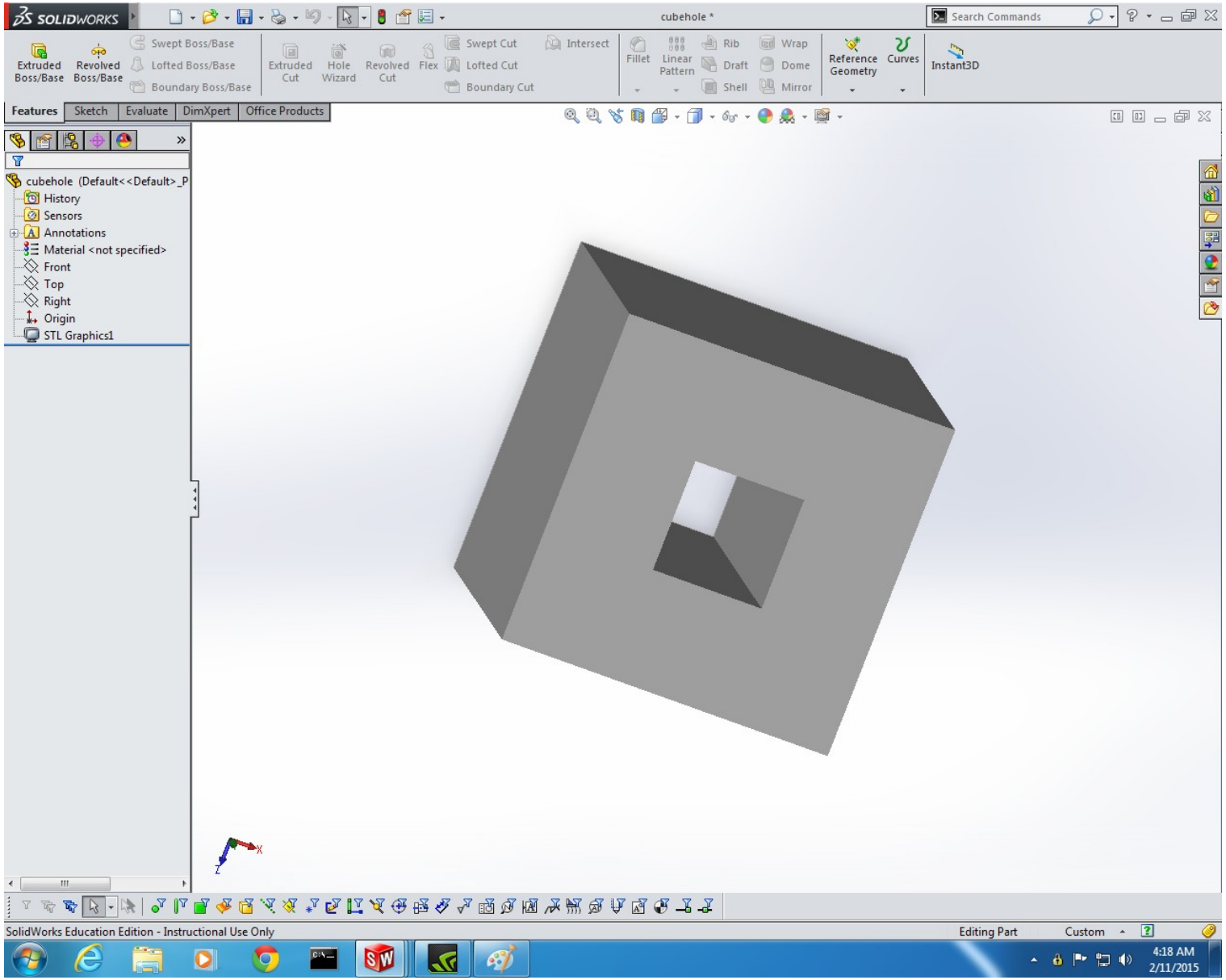
With Zero Surface Normals



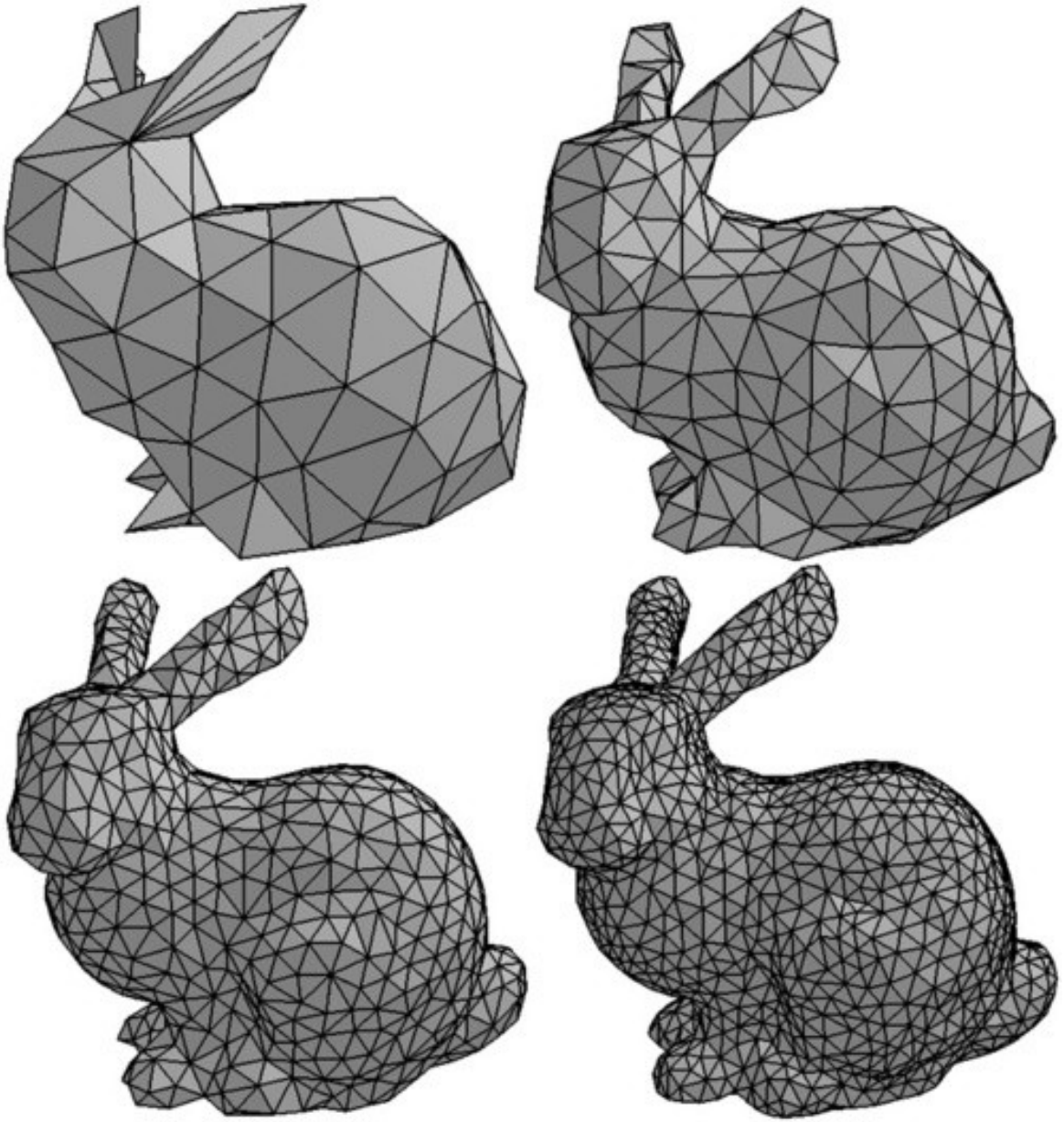
With Proper Surface Normals



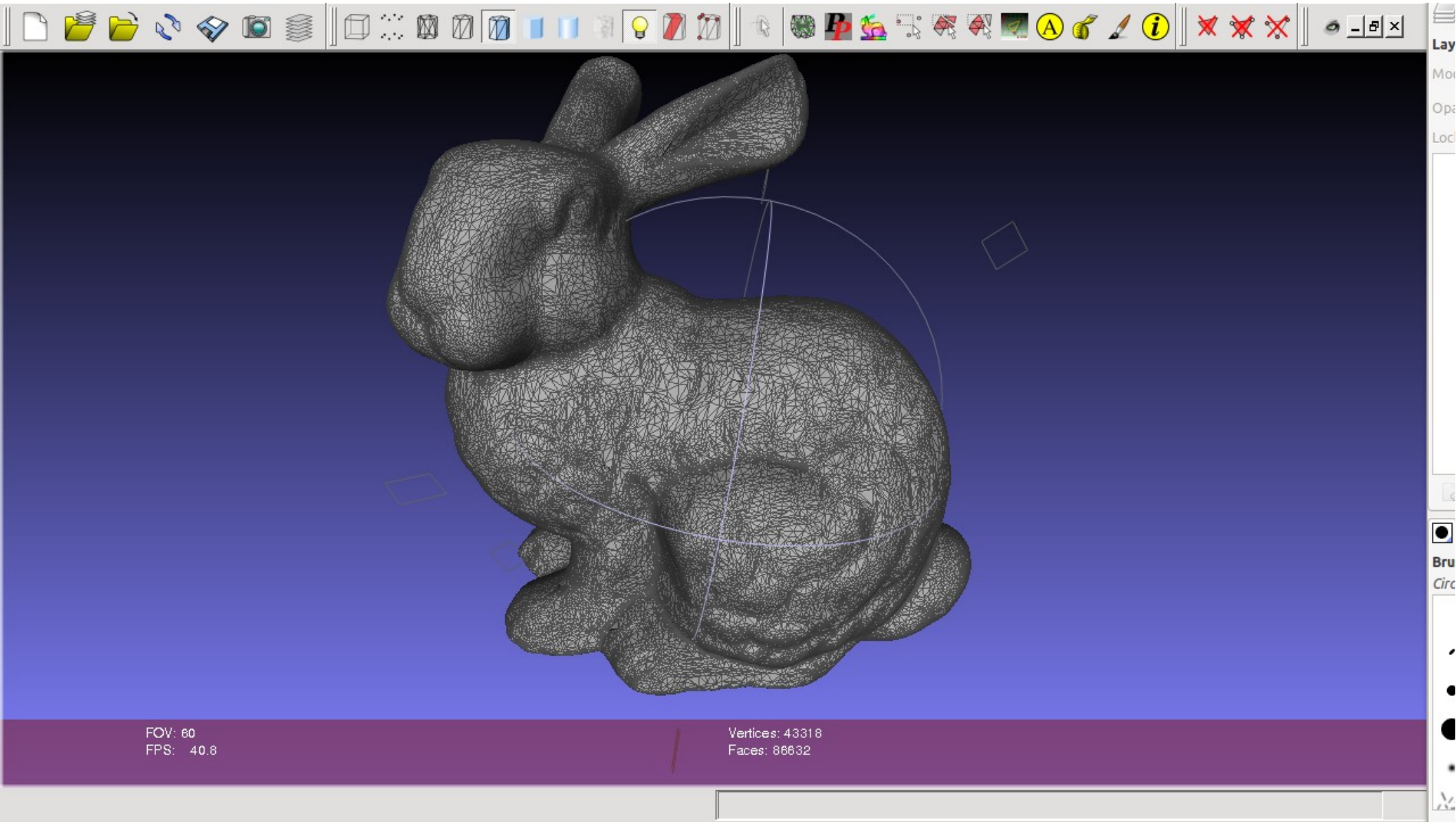
Cube With a Hole In It



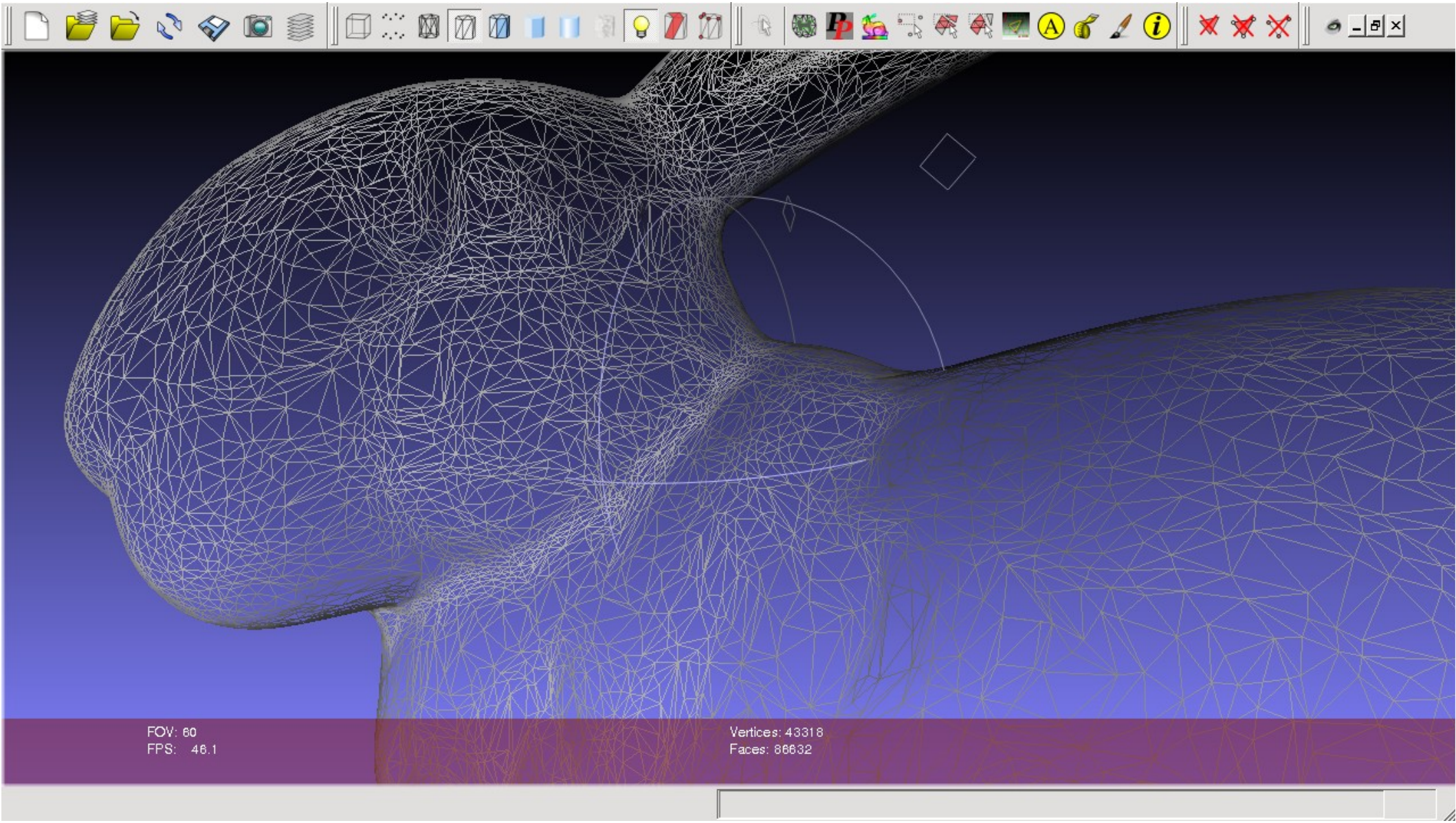
The Stanford Bunny: Low Res



MeshLab: Hi-Res Bunny

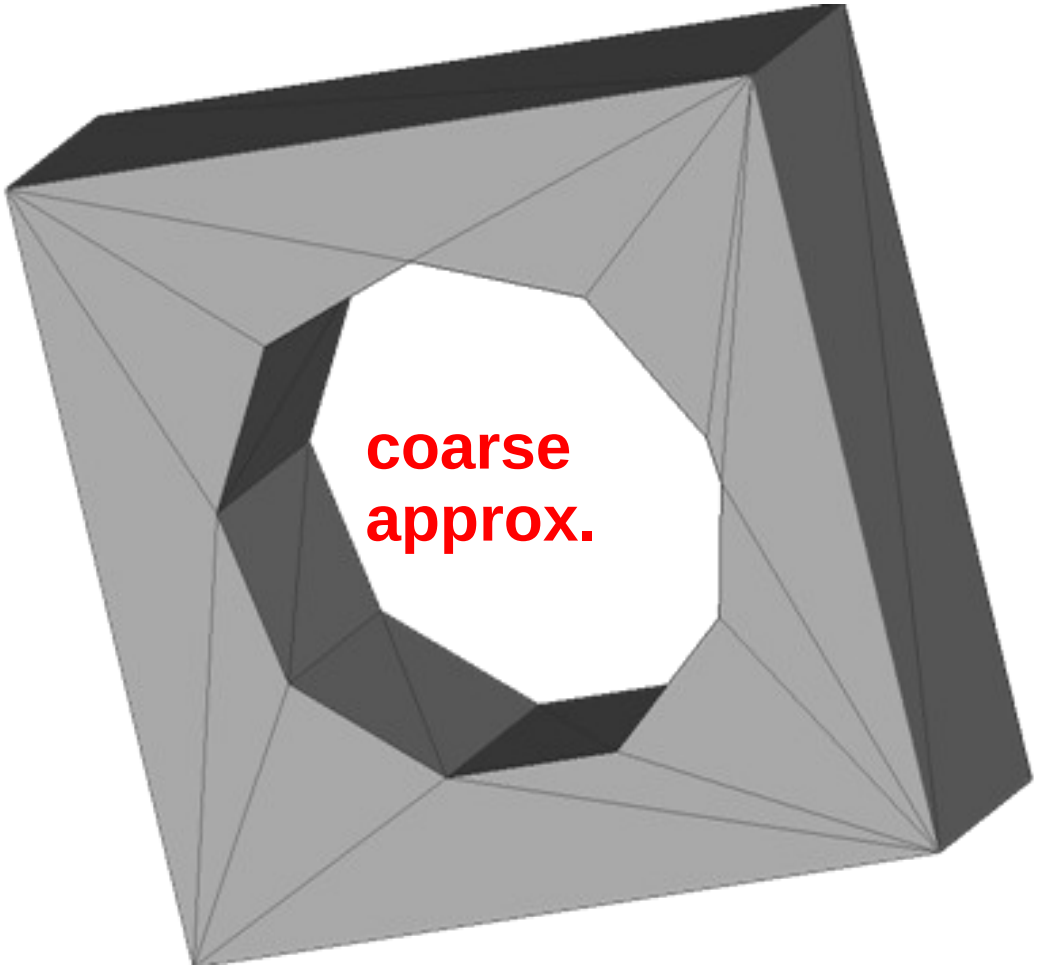


Zooming In with MeshLab



Triangulation: Resolution 0.1 Inches

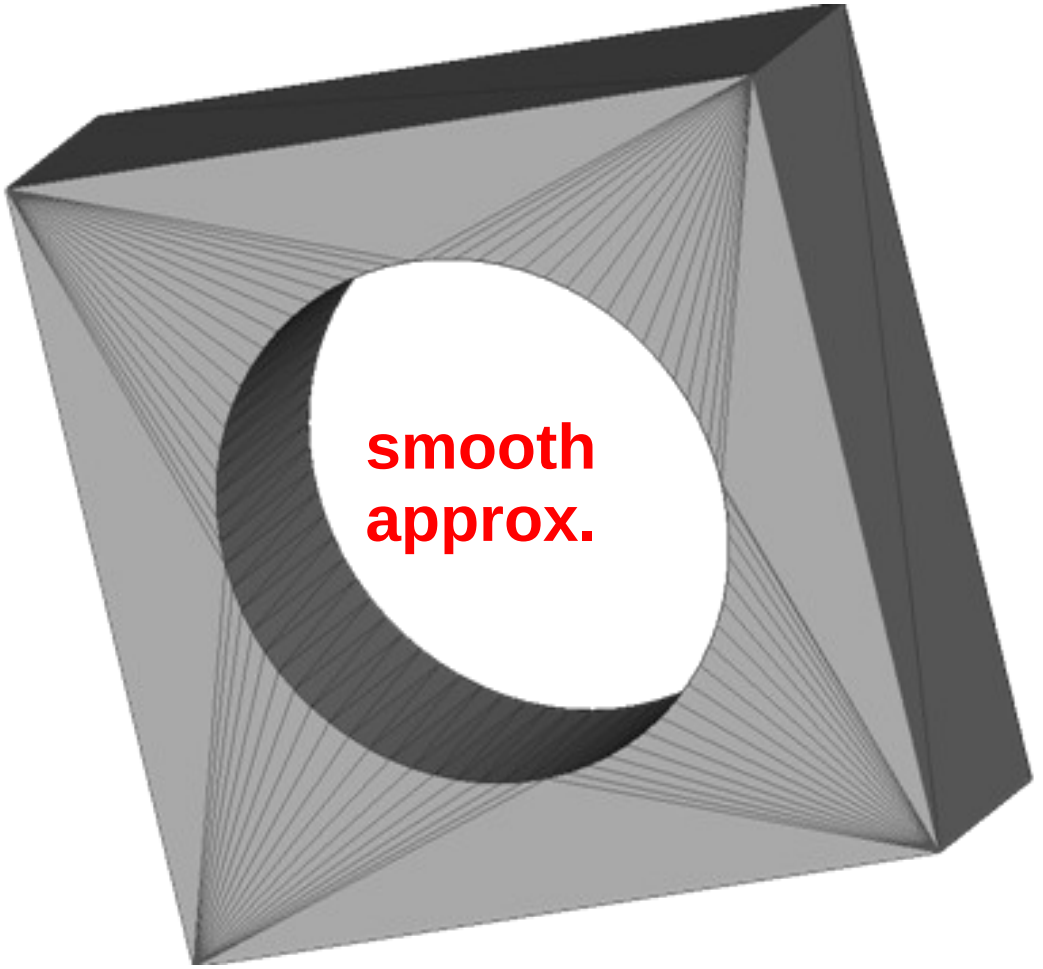
2 inch round hole



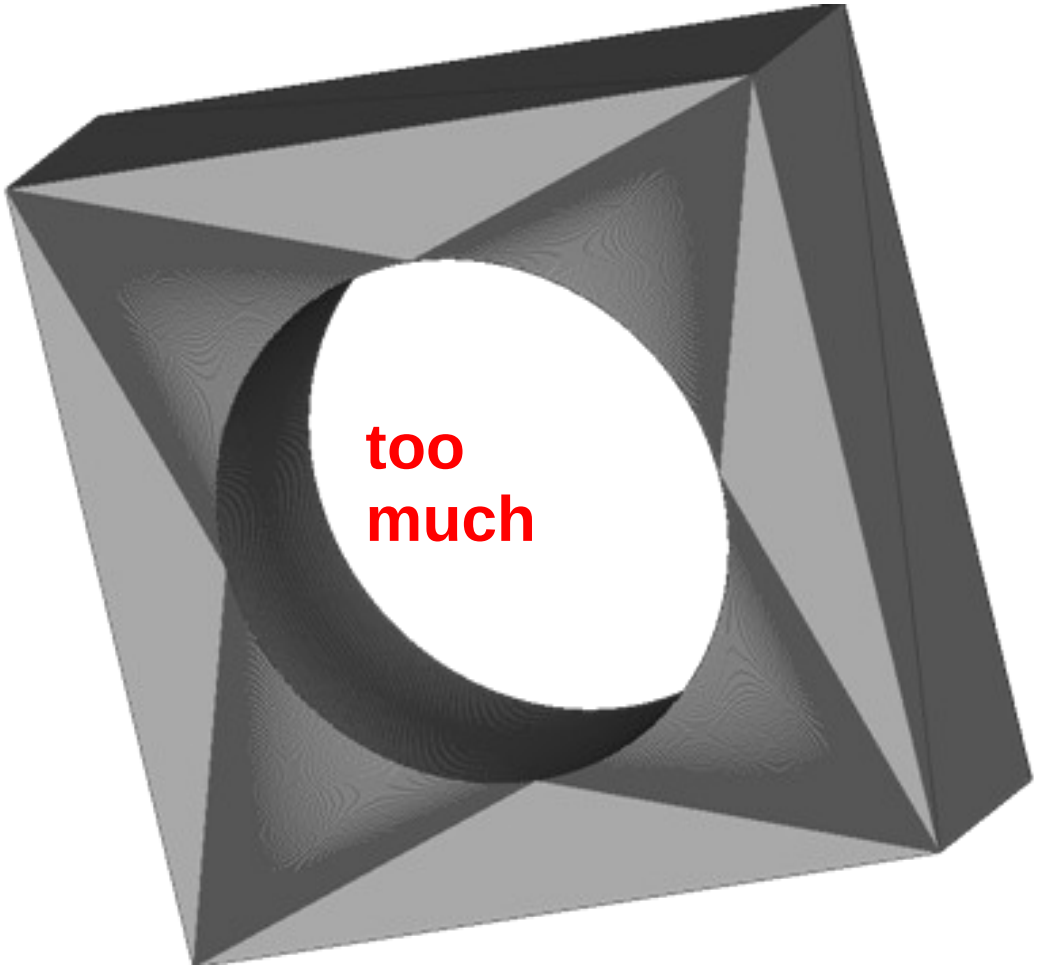
coarse approx.

Resolution 0.01 Inches

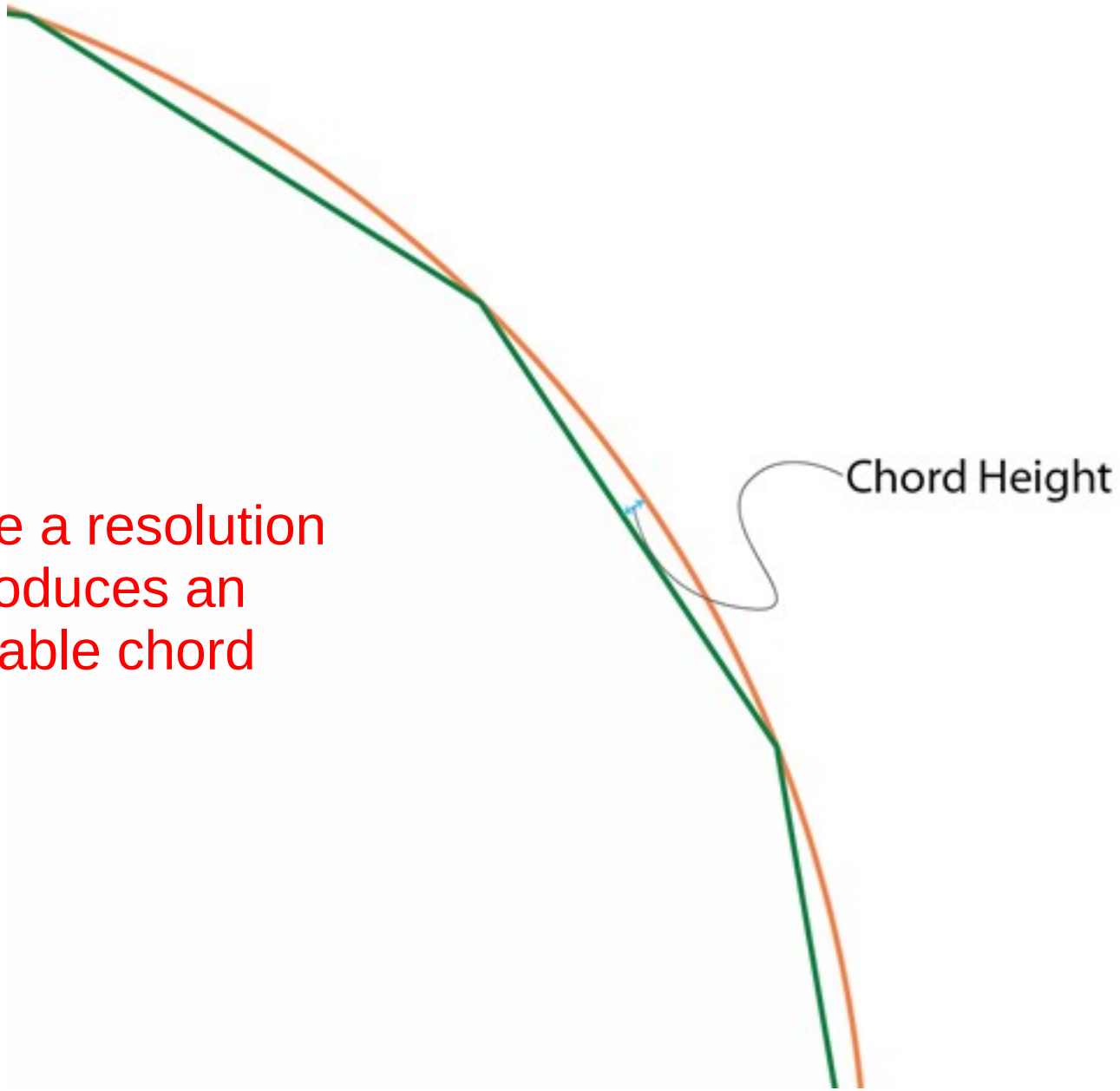
Resolution 0.001 Inches



Resolution 0.0001 Inches



Chord Height = Max Distance from Actual Surface to the Facet

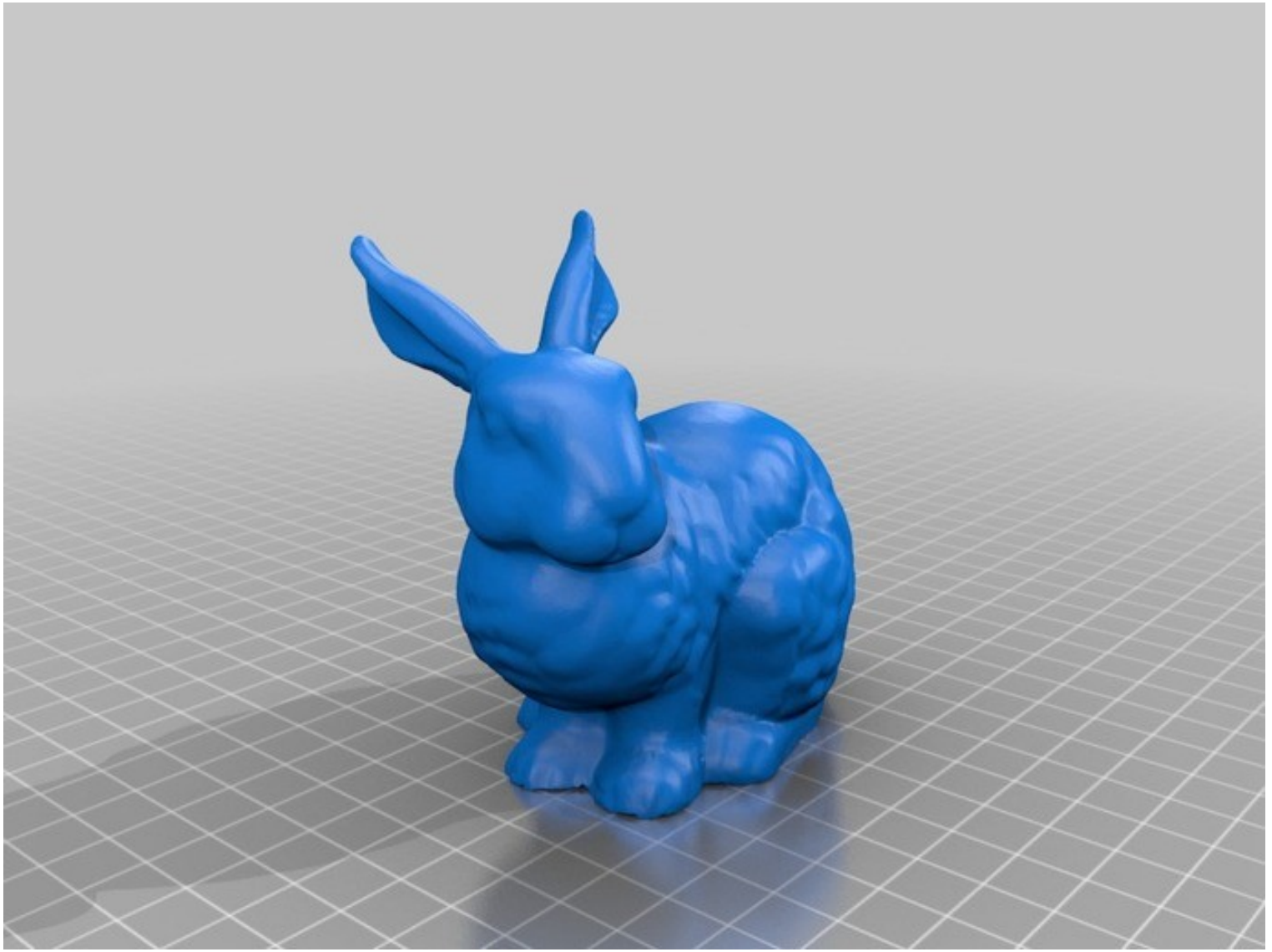


Choose a resolution that produces an acceptable chord height.

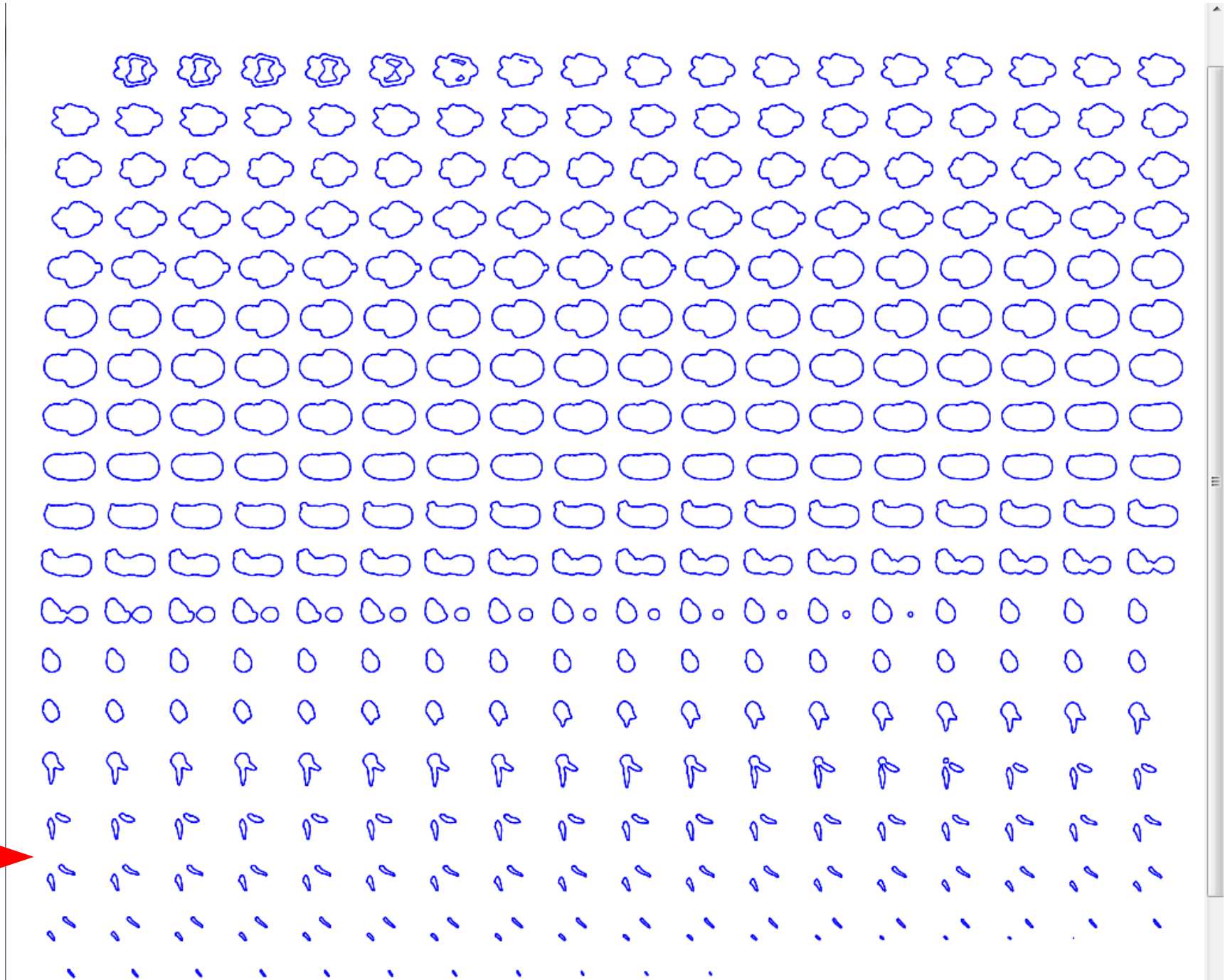
3D Printing Tool Chain (Simplified)

- Shell the object.
- Add infill (internal lattice) for strength.
- Add supports and raft if requested.
- Slice the object into layers.
- For each layer, compute a “tool path” for the extruder to follow.

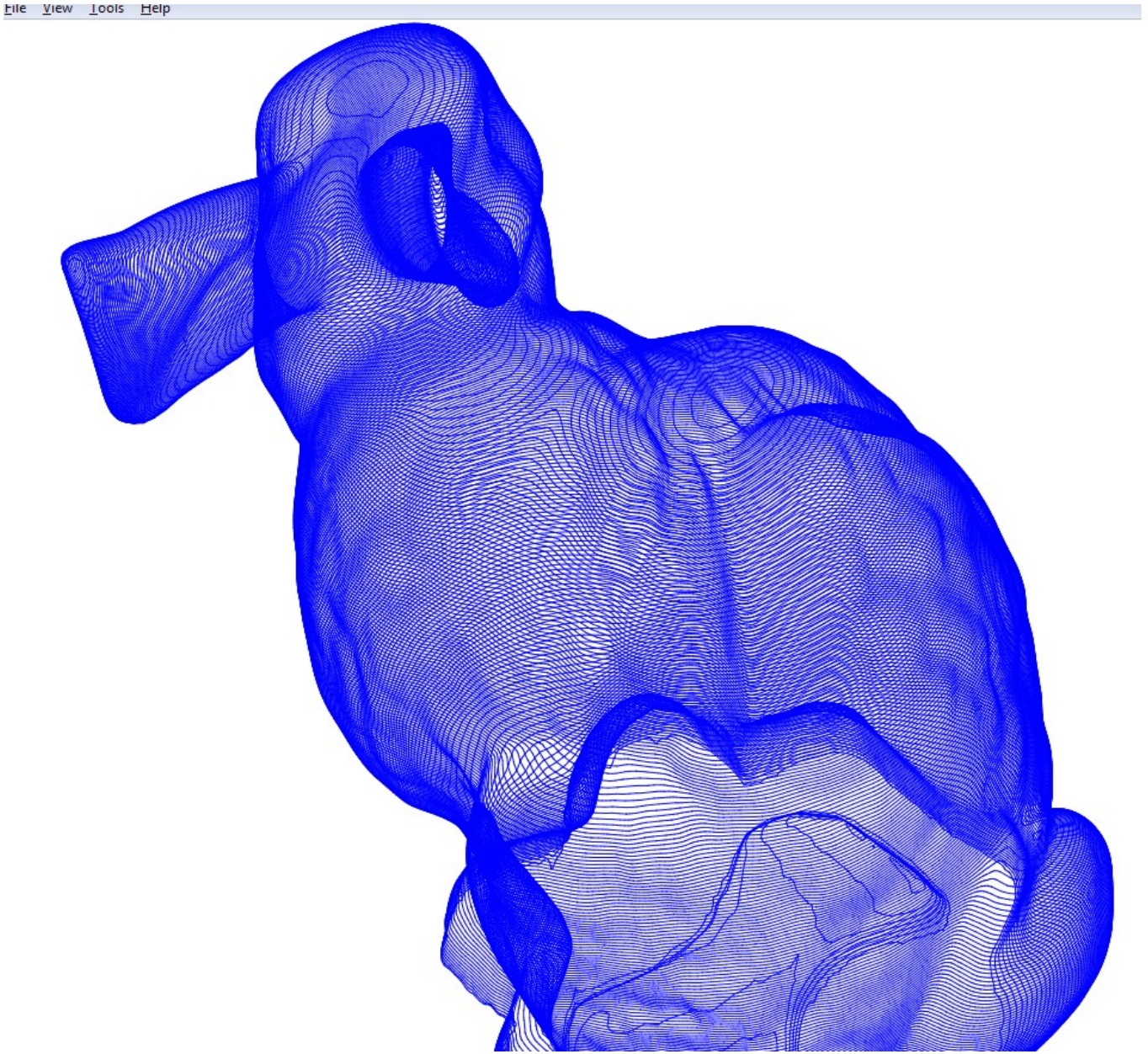
Slicing the Bunny



Bunny Slice Outlines



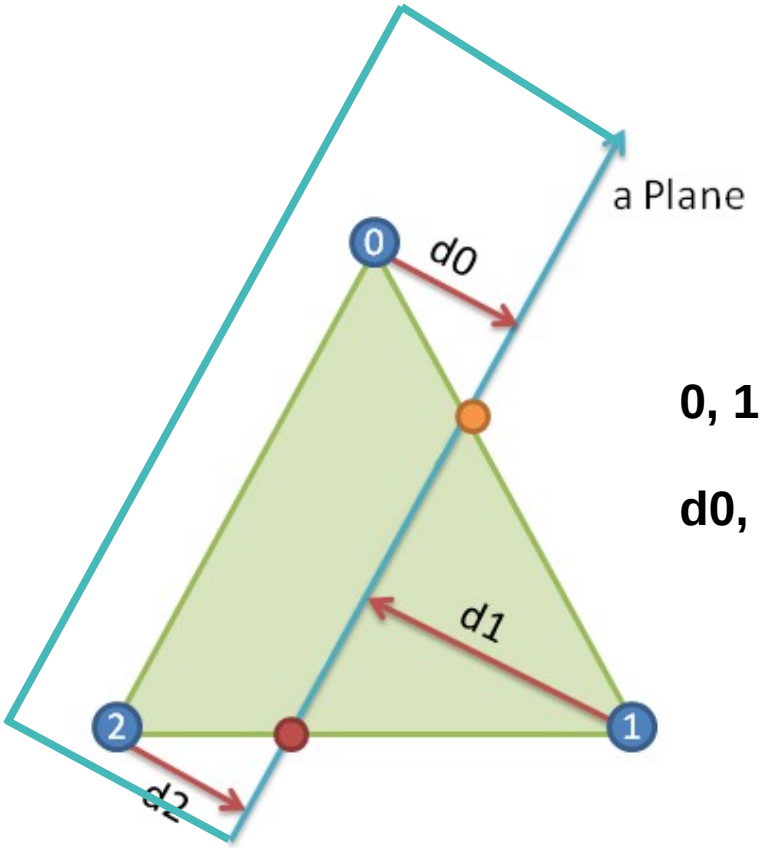
Sliced Bunny



Slicing Algorithm

- Given the cutting plane orientation and the bounding box of the object, determine the number of slices (cutting planes).
- For each triangle in the mesh:
  - For each cutting plane:
    - Compute the intersection of the cutting plane and the triangle.
    - If the intersection contains exactly 2 points, add that line to the list of line segments for that cutting plane.
- For each cutting plane:
  - Assemble the list of line segments to form a set of continuous lines. These will be converted to tool paths.

Does the Triangle Intersect the Cutting Plane?



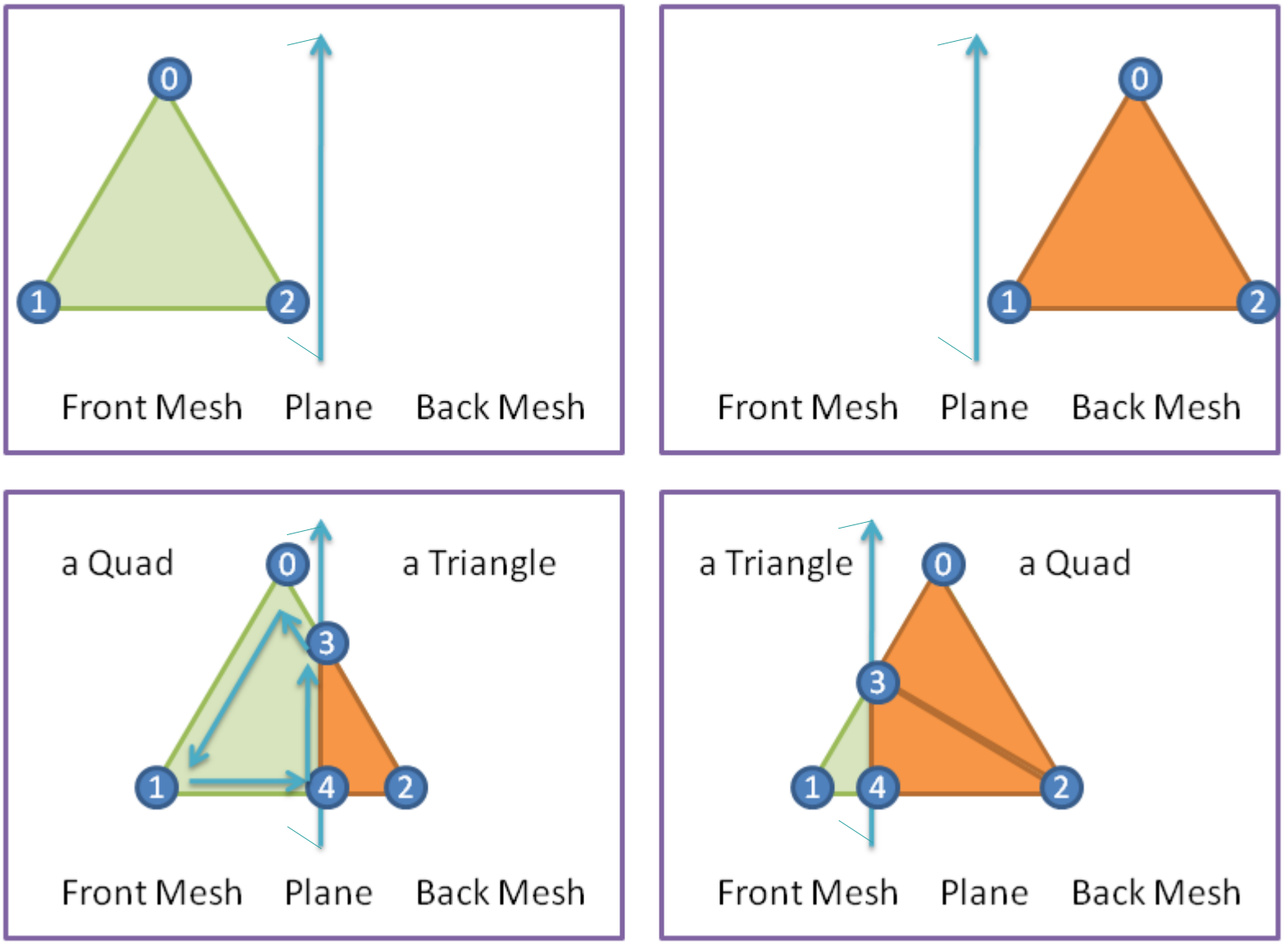
0, 1, 2 – triangle vertices

d0, d1, d2 – distance of vertex from cutting plane

A **Plane** has {v3 normal, float distance}
Distance from vertex to plane: vertex.dotproduct(plane.normal) – plane.distance

Figure from ravehgonen.wordpress.com.

Triangle Slicing – 4 Cases



* There are more degenerate cases (The plane "falls" on one of the original vertices –>no quad generated)

Figure from ravehgonen.wordpress.com.

Algorithm Outline

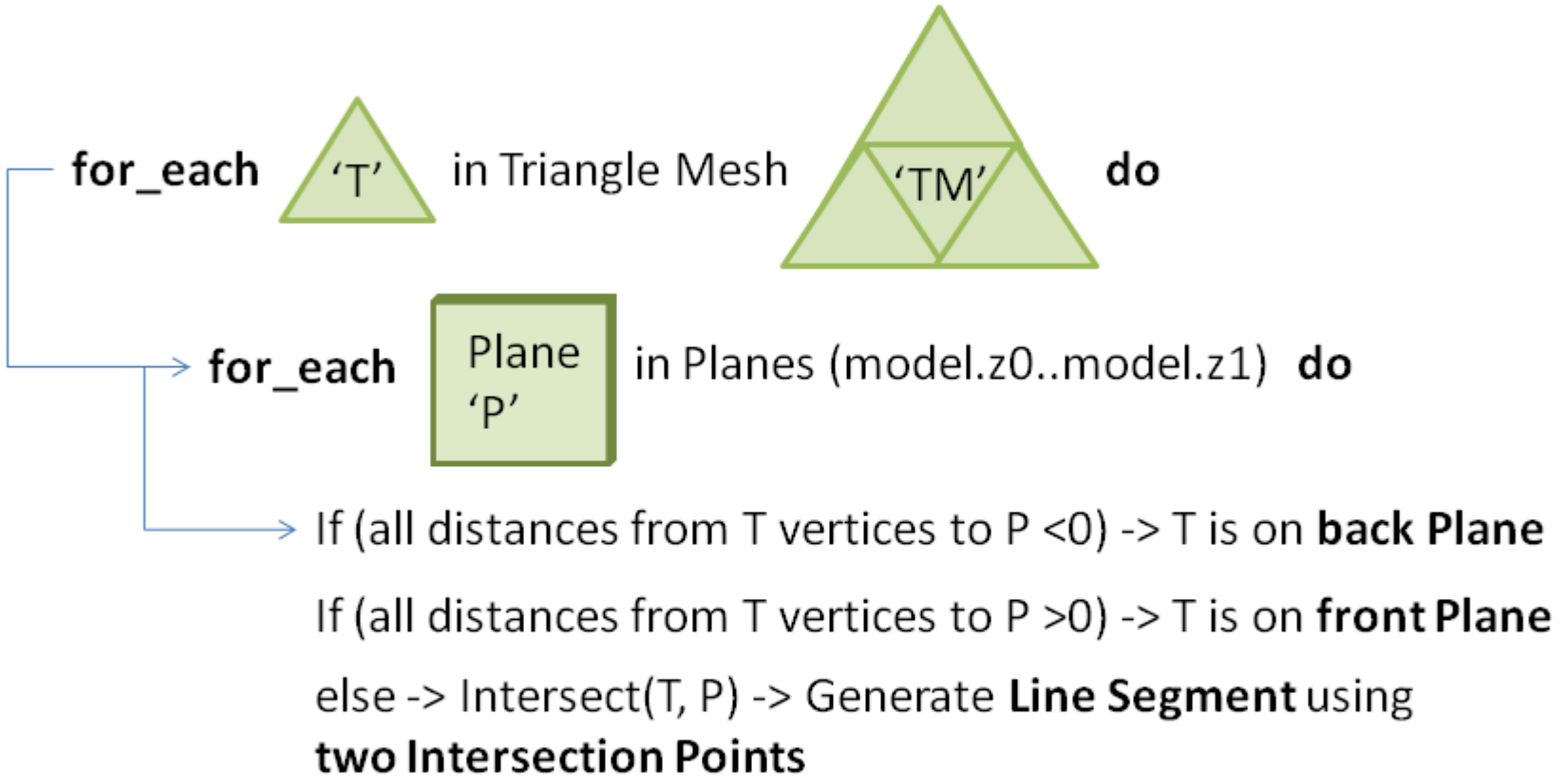
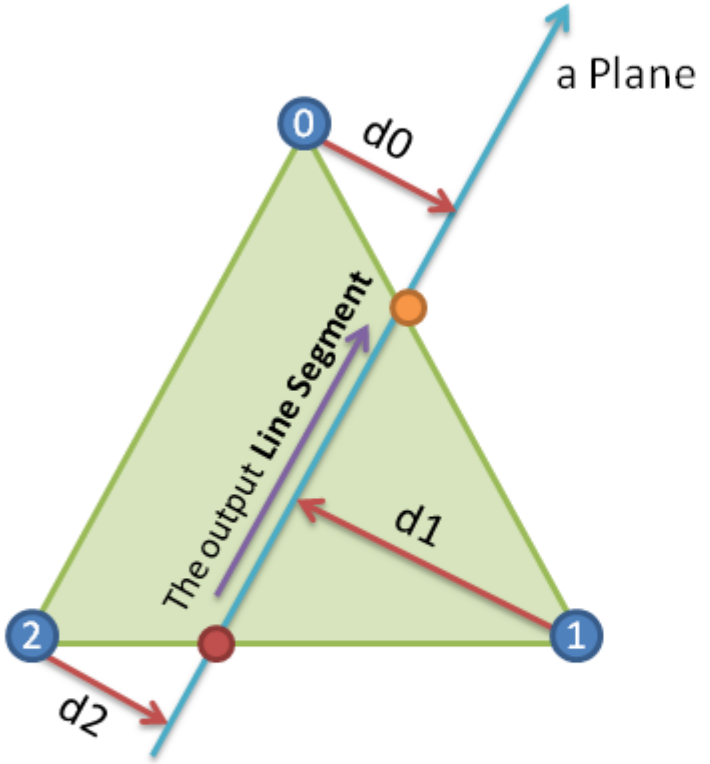


Figure from ravehgonen.wordpress.com.

A **Plane** has {v3 normal, float distance}
Distance from vertex to plane: vertex.dotproduct(plane.normal) – plane.distance

Finding Triangle Plane Intersection



If (d0*d1<0)

s10 = d1/(d1-d0) s21 = d2/(d2-d1)

Intersection points = LinearInterp(1, 0, s10)

Intersection points = LinearInterp(2, 1, s21)

Output line segment

Figure from ravehgonen.wordpress.com.

A **Plane** has {v3 normal, float distance}

Distance from vertex to plane: vertex.dotproduct(plane.normal) – plane.distance

GCcode

- The output of the slicer program is typically a GCode file.
- GCode is used in many types of CNC machines. (CNC = Computerized Numerical Control)
- Includes commands to move the extruder to specified (x,y,z) coordinates, feed (or stop feeding) plastic, etc.

Popular Slicing Programs

- Slic3r
- Cura
- KISSlicer
- Skeinforge