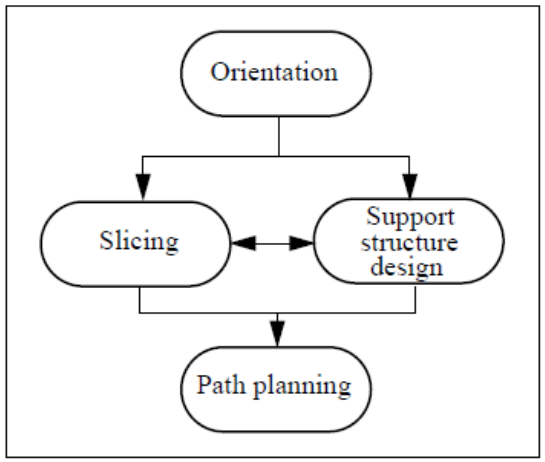
3D Printing Software Pipeline

Sai-Kit Yeung

ISTD, SUTD

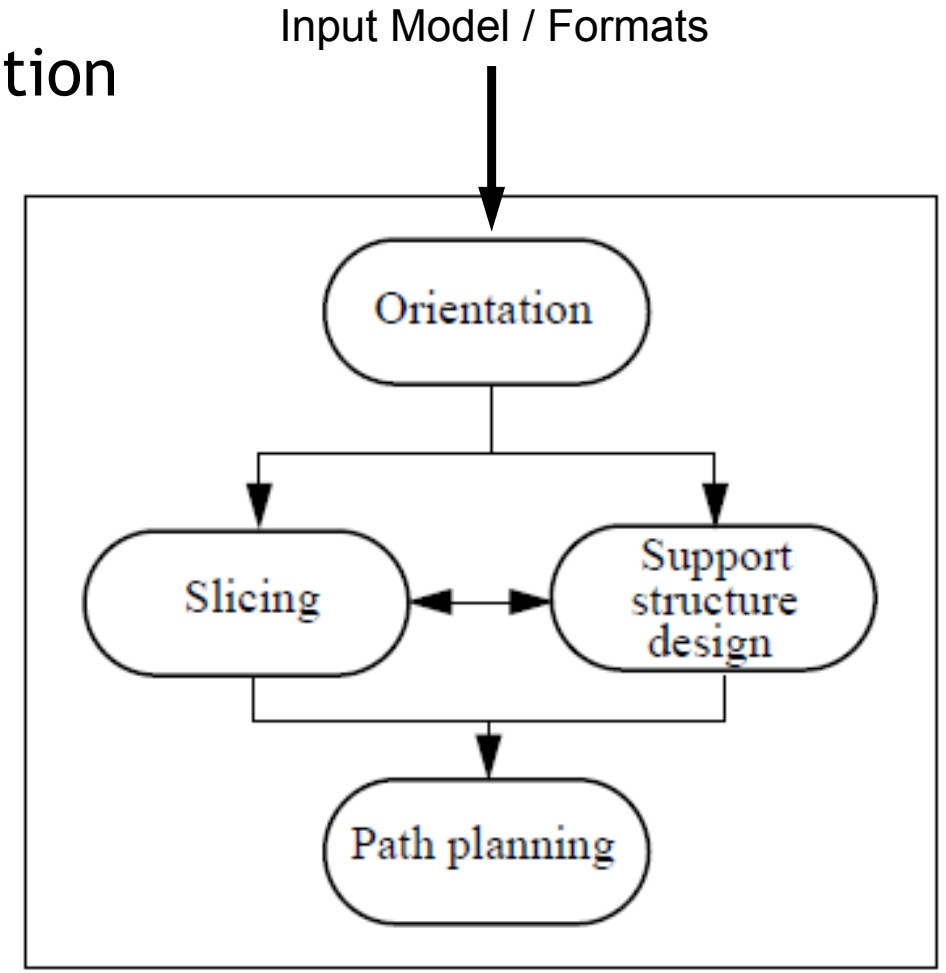
Plan for Today

- Process Planning
  - Input Model Formats
  - Orientation Determination
  - Support Structure Determination
  - Slicing
  - Path Planning
  - Machine Instructions



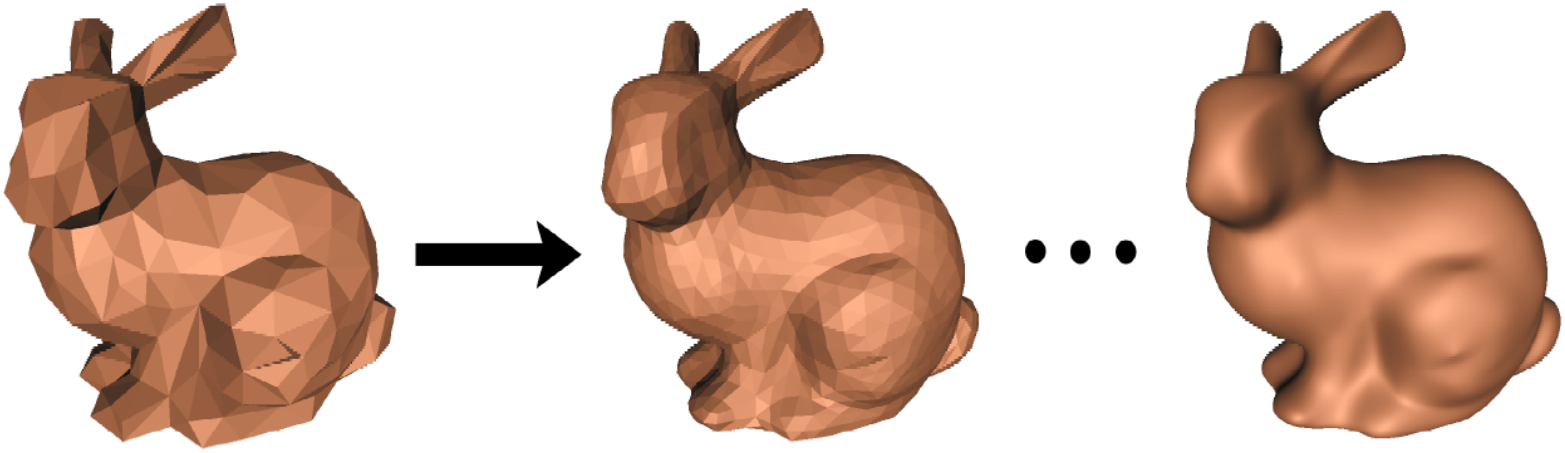
Process Planning

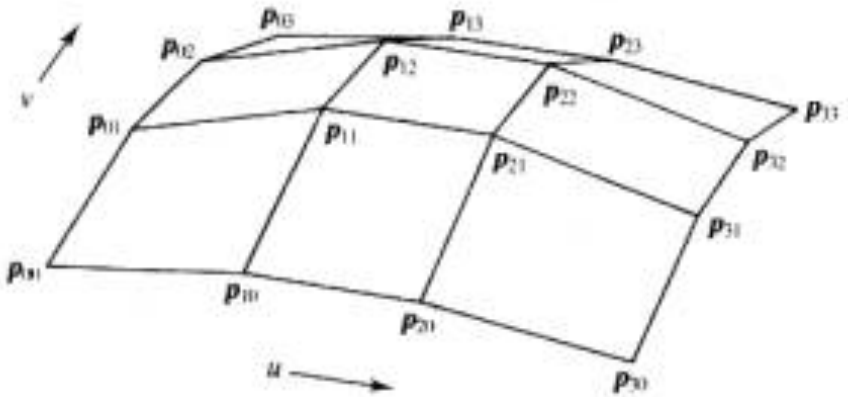
- Input Model Formats
- Orientation Determination
- Support Structure Determination
- Slicing
- Path Planning
- Machine Instructions

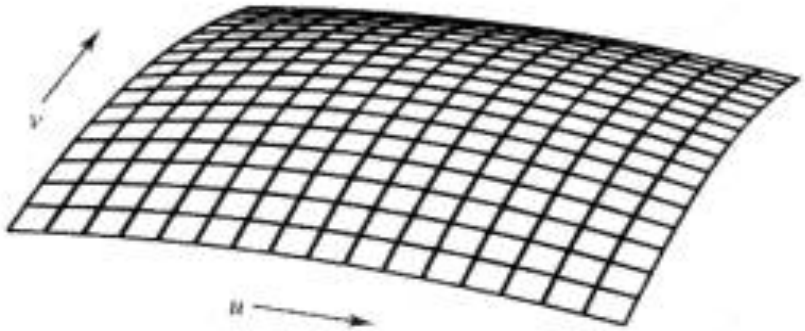


Specifying Input Models

- Surface models
  - triangles, subdivision surfaces, free-form surfaces (NURBS)

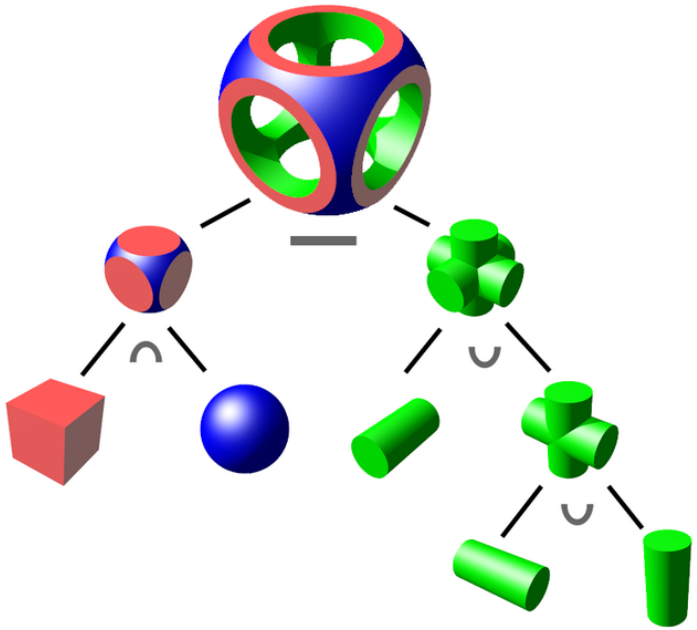


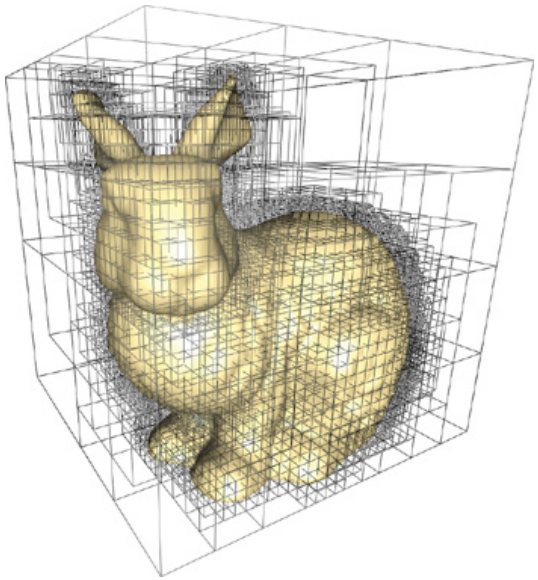


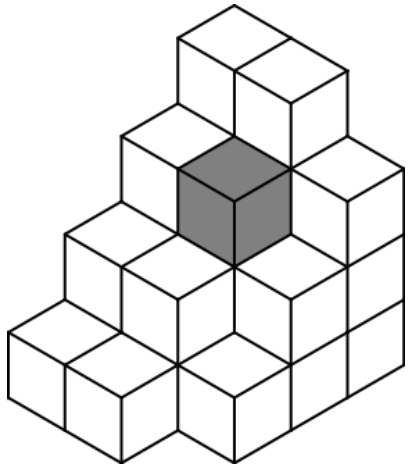


Specifying Input Models

- Solid models
  - Boundary representations (B-rep), constructive solid geometry (CSG), voxels, octrees



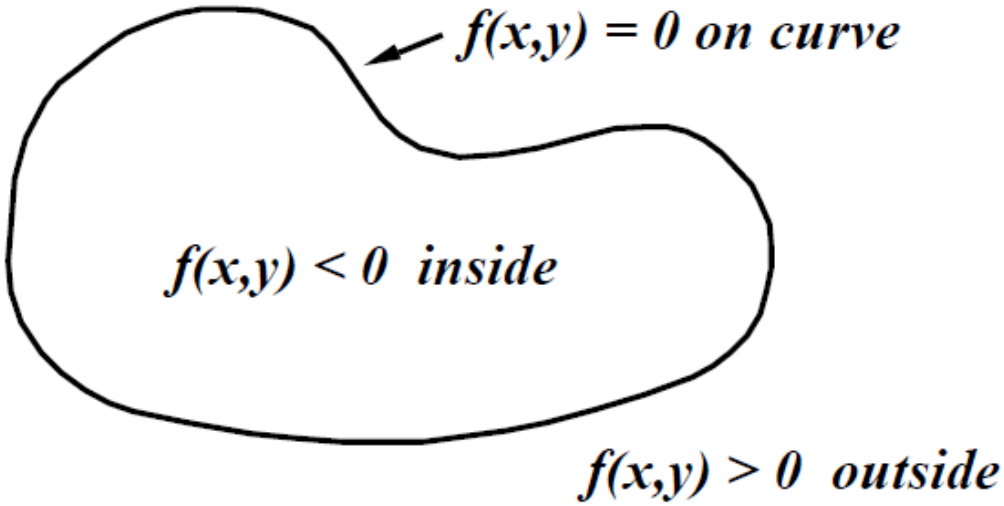


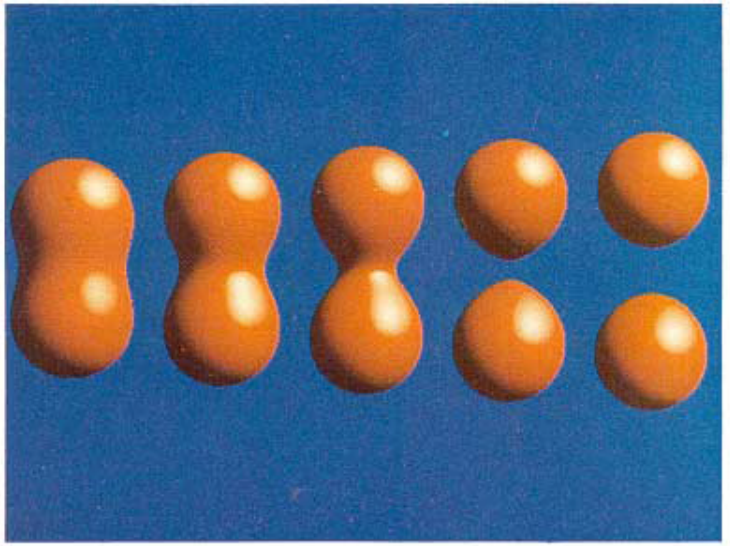


Specifying Input Models

- Implicit surfaces – surface defined implicitly by a function

f (x, y, z) = 0 (on surface)
f (x, y, z) < 0 (inside)
f (x, y, z) > 0 (outside)





Specifying Input Models

- Point sets / point set surfaces

Alexa et al. 2001

STL (Stereolithography) File Format

- Standard Tessellation Language, Developed by 3D Systems
- Triangle “soup” – an unordered list of triangular facets
- Vertices ordered by the right hand rule

ASCII

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

} repeat for each triangle

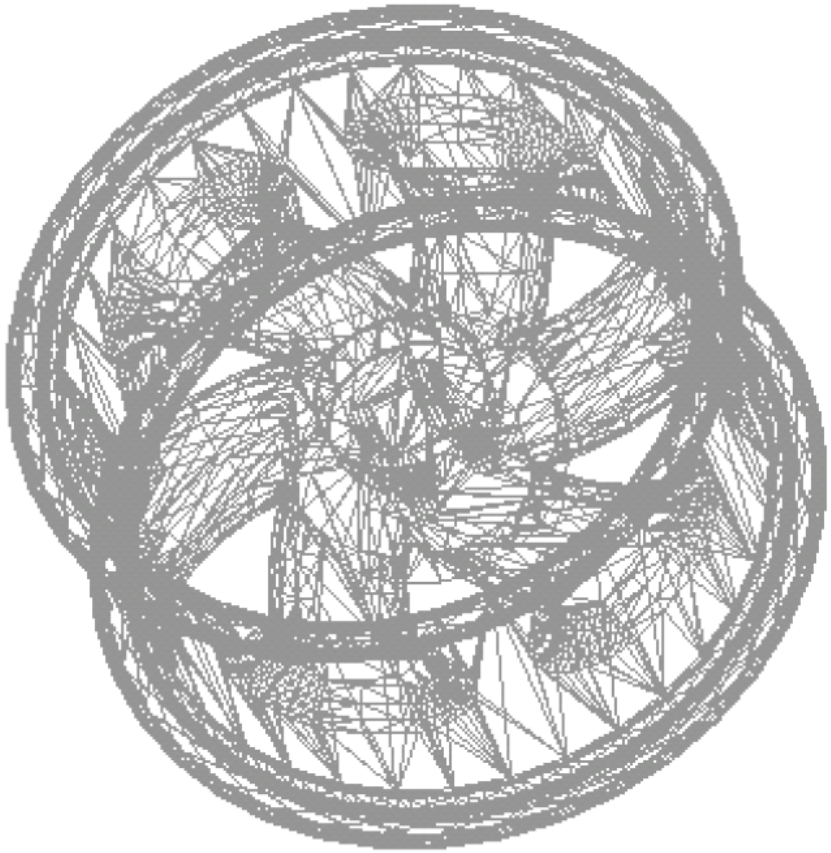
endsolid *name*

binary

UINT8[80] – Header
UINT32 – Number of triangles

foreach triangle
REAL32[3] – Normal vector
REAL32[3] – Vertex 1
REAL32[3] – Vertex 2
REAL32[3] – Vertex 3
UINT16 – Attribute byte count (0)
end

STL (Stereolithography) File Format

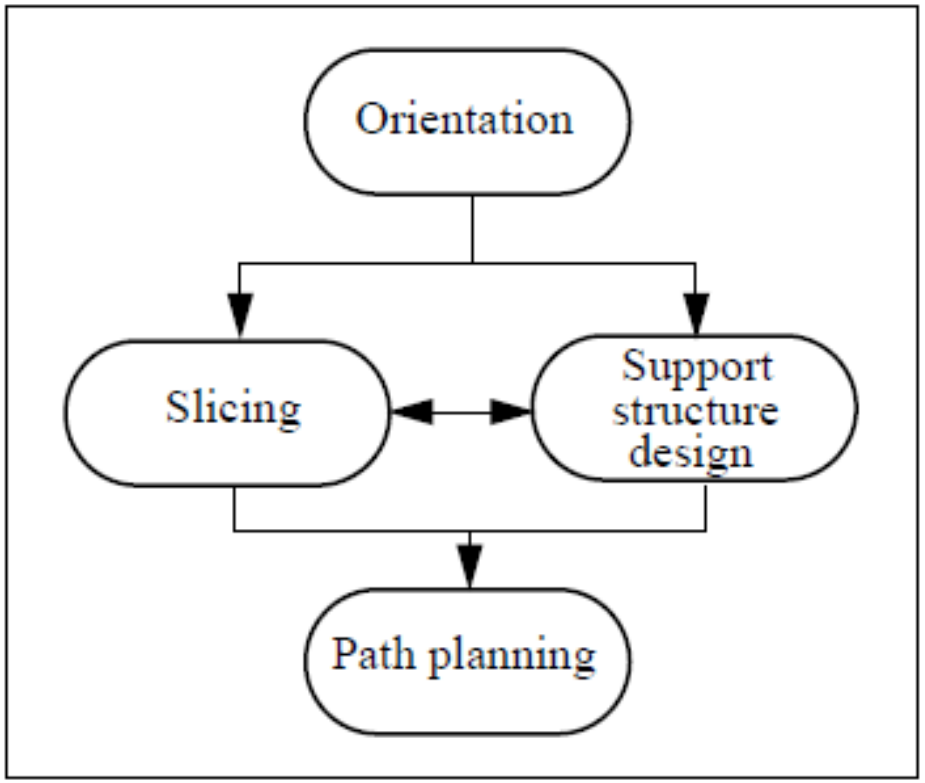


```
solid Wheel
   facet normal -1.000000e+000 0.000000e+000 0.000000e+000
      outer loop
         vertex 7.095000e+001 2.913194e+002 7.026579e+001
         vertex 7.095000e+001 2.914028e+002 7.636772e+001
         vertex 7.095000e+001 3.106206e+002 8.149973e+001
      endloop
   endfacet
   facet normal -1.000000e+000 0.000000e+000 0.000000e+000
      outer loop
         vertex 7.095000e+001 3.106206e+002 8.149973e+001
         vertex 7.095000e+001 2.914028e+002 7.636772e+001
         vertex 7.095000e+001 2.882984e+002 1.048139e+002
      endloop
   endfacet
   facet normal -1.000000e+000 0.000000e+000 0.000000e+000
      outer loop
         vertex 7.095000e+001 3.106206e+002 8.149973e+001
         vertex 7.095000e+001 2.882984e+002 1.048139e+002
         vertex 7.095000e+001 2.795565e+002 1.320610e+002
      endloop
   endfacet
   facet normal -1.000000e+000 0.000000e+000 0.000000e+000
      outer loop
         vertex 7.095000e+001 2.685262e+002 2.101446e+002
         vertex 7.095000e+001 2.845330e+002 1.940968e+002
         vertex 7.095000e+001 2.647845e+002 1.974923e+002
      endloop
                .
                .
                .
   facet normal -1.000000e+000 0.000000e+000 0.000000e+000
      outer loop
         vertex 7.095000e+001 2.647845e+002 1.974923e+002
         vertex 7.095000e+001 2.845330e+002 1.940968e+002
         vertex 7.095000e+001 3.011244e+002 1.720122e+002
      endloop
   endfacet
endsolid
```

Source: Jackson 2000

Process Planning

- Input Model Formats
- Orientation Determination
- Support Structure Determination
- Slicing
- Path Planning
- Machine Instructions

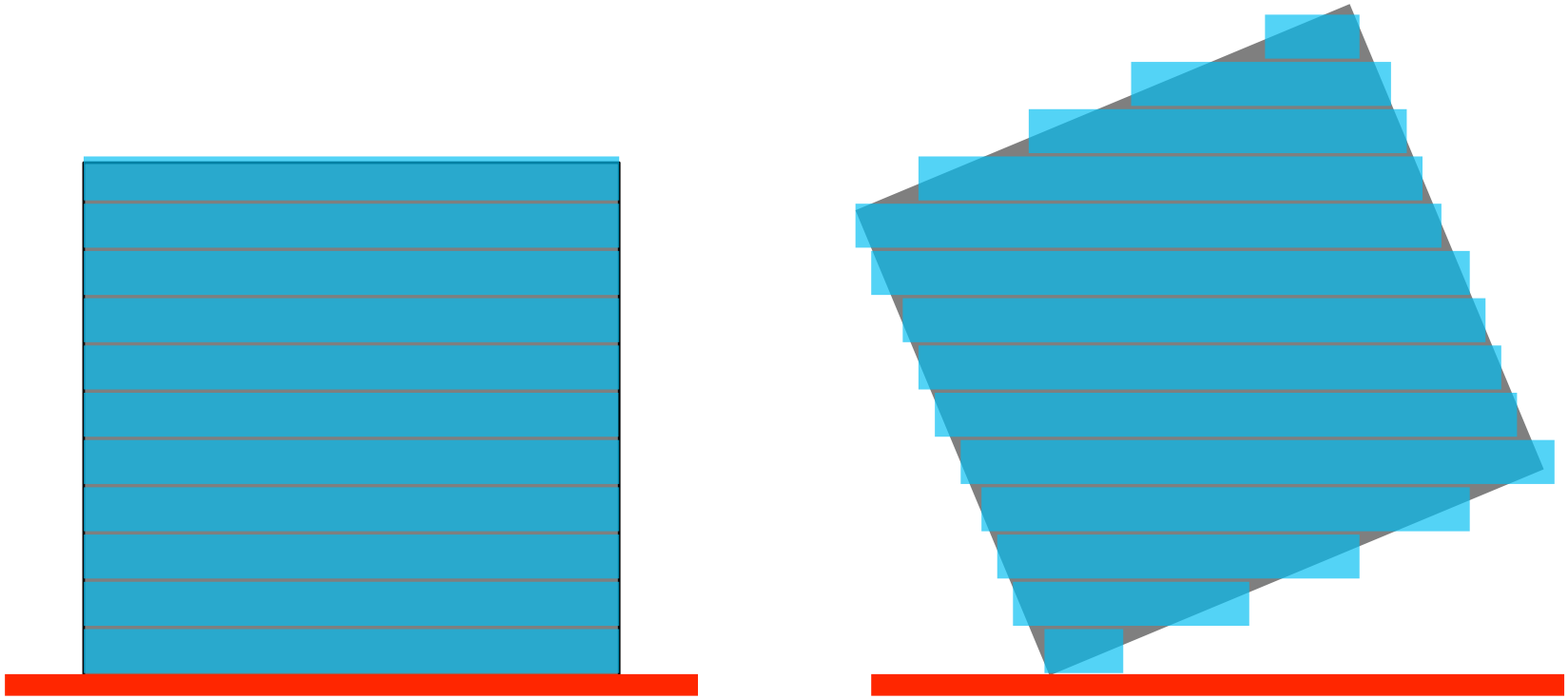


Orientation Determination

- In which orientation the part will be built
- Factors
  - Surface accuracy
  - Build time
  - Support volume
  - Support contact area
  - Mechanical properties
- Involves analysis of the 3D model

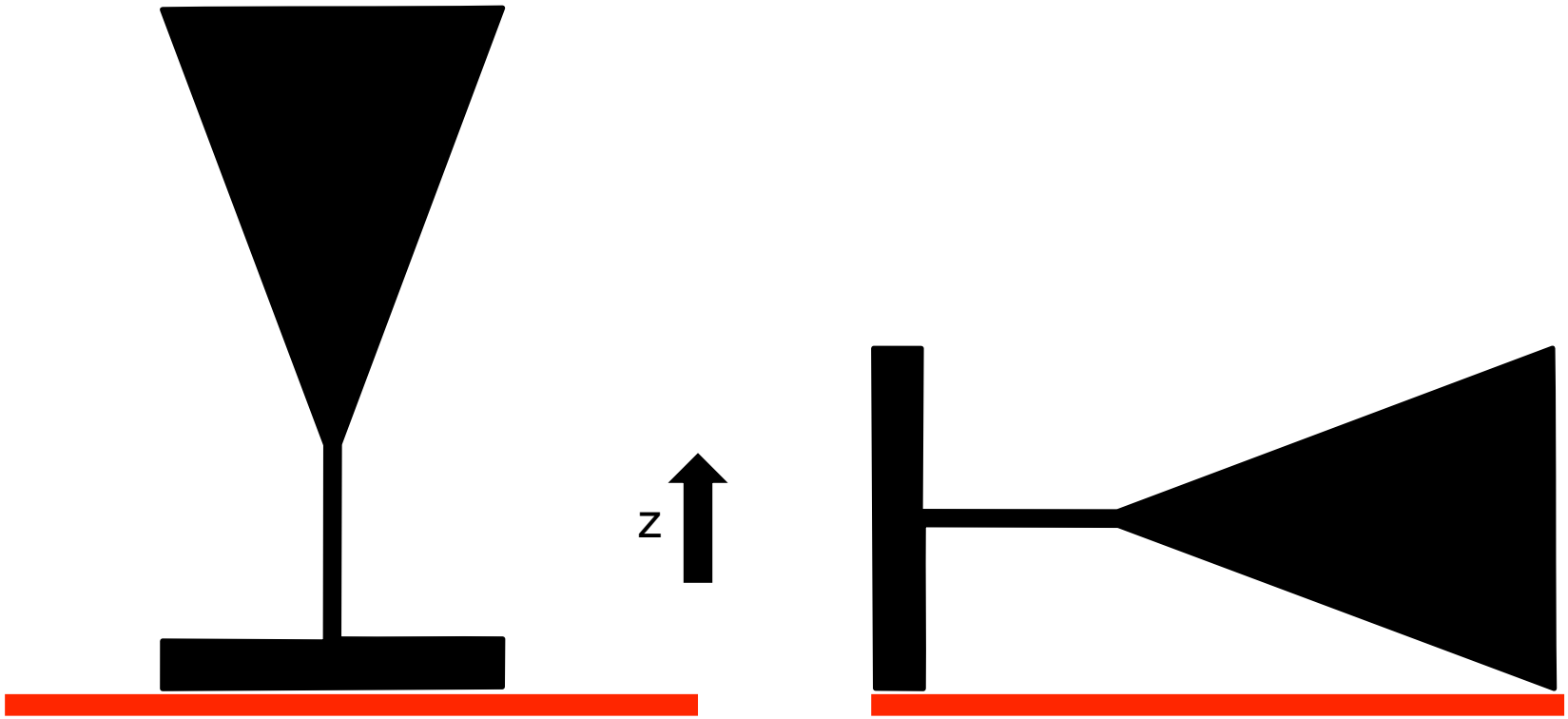
Orientation Determination: Surface Accuracy

- Difference between the input shape and the printed shape
  - Difference in volume
  - Difference in the normal direction



Orientation Determination: Build Time

- Build speed is slower for the z direction compared to the xy direction
- Build time can be computed by simulating the device

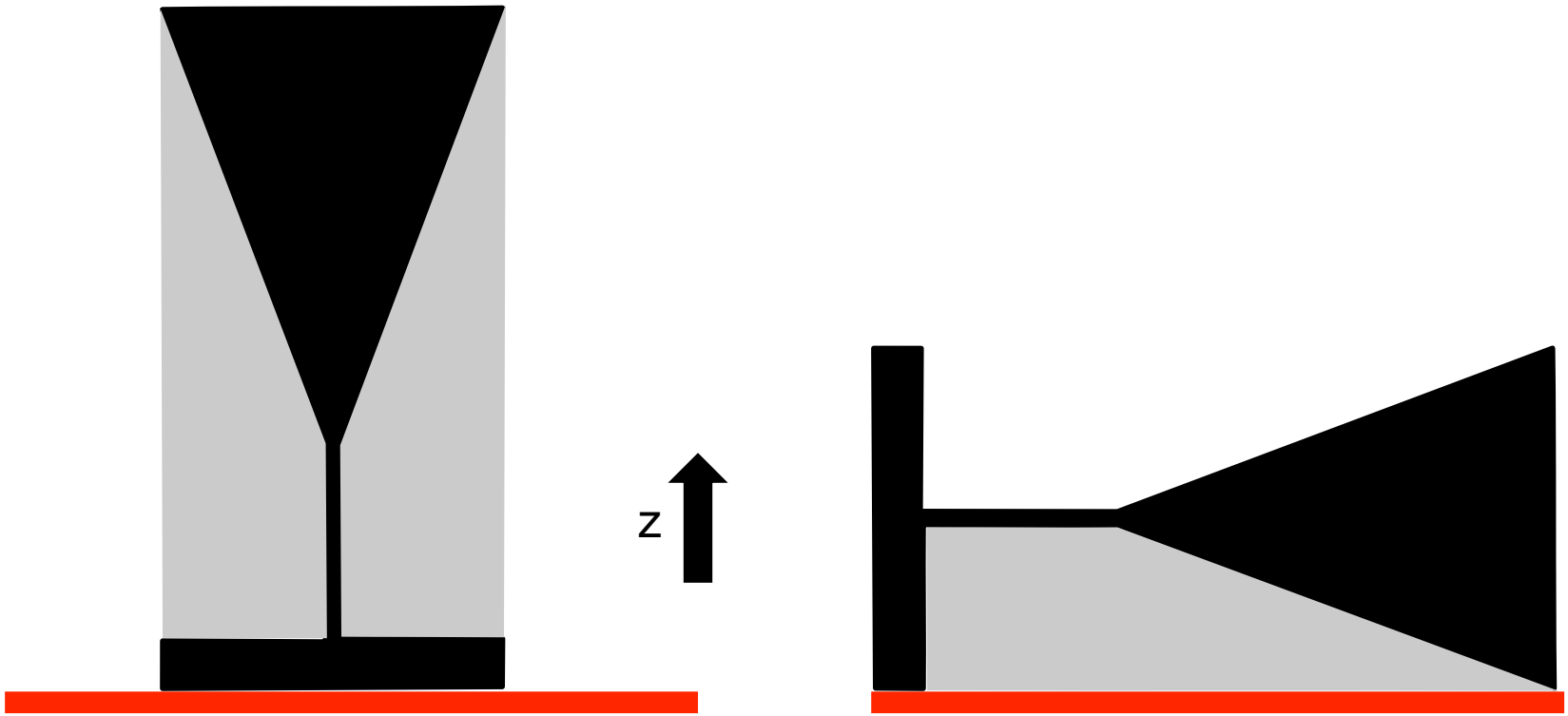


longer build time

shorter build time

Orientation Determination: Support Volume

- Support material volume should be minimized
- Support material volume can be computed geometrically

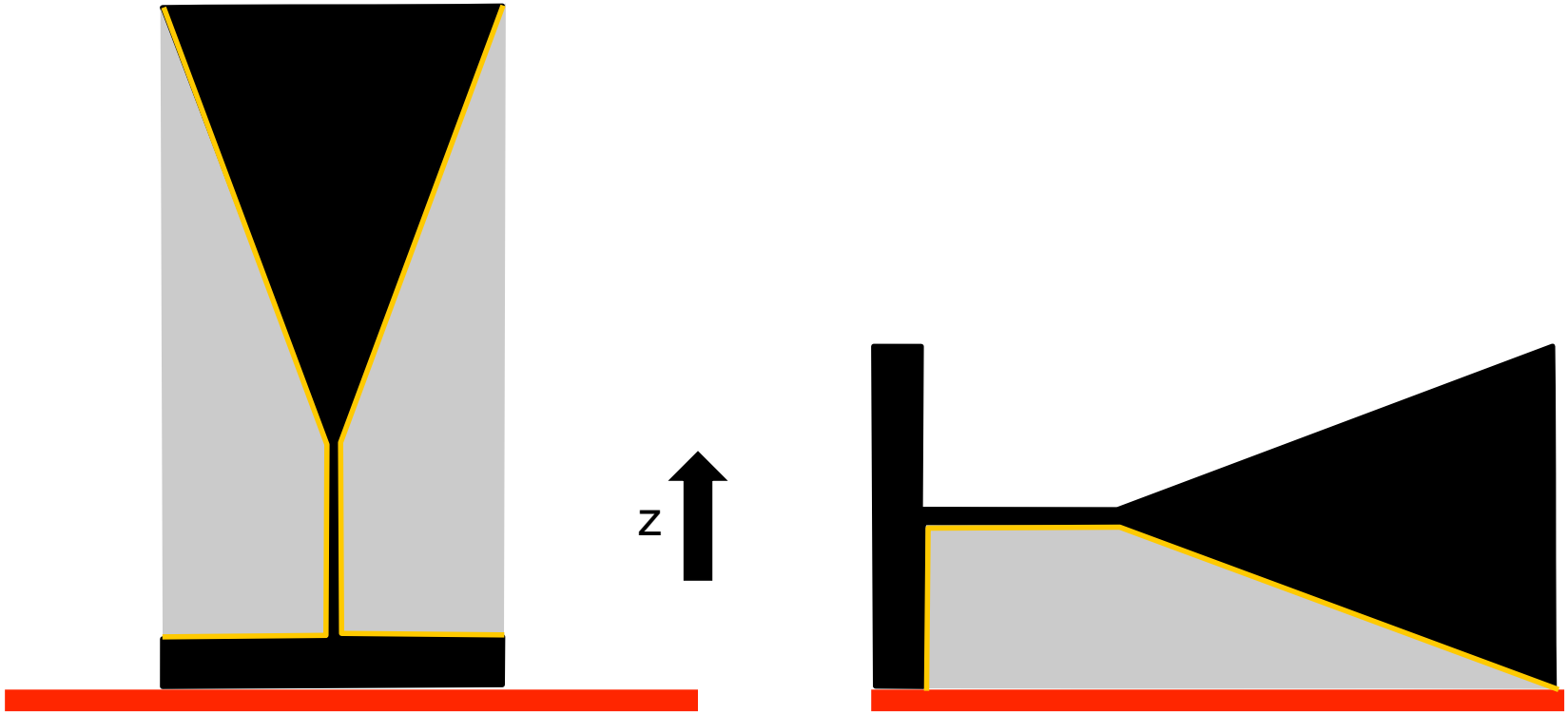


more support volume

less support volume

Orientation Determination: Support Contact Area

- Contact area between support material and object
  - should be minimized
  - can be geometrically computed

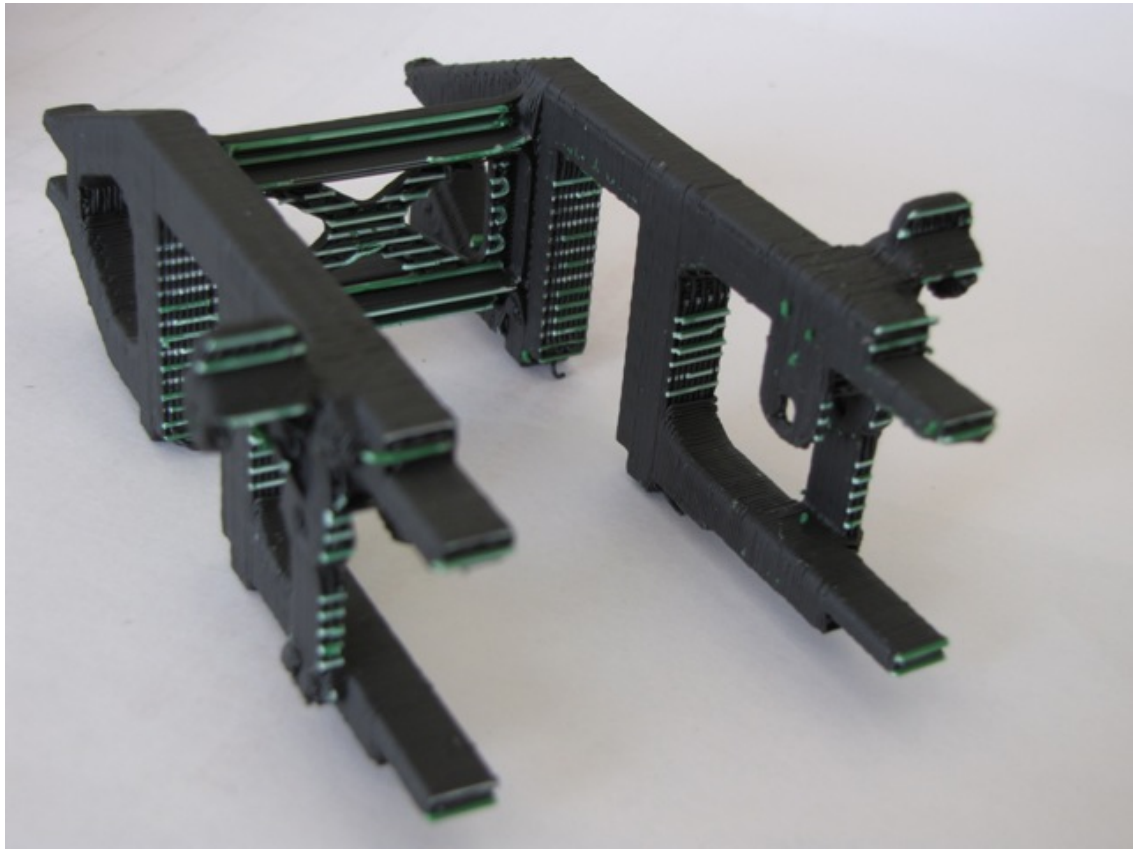


larger contact area

smaller contact area

Orientation Determination: Support Contact Area

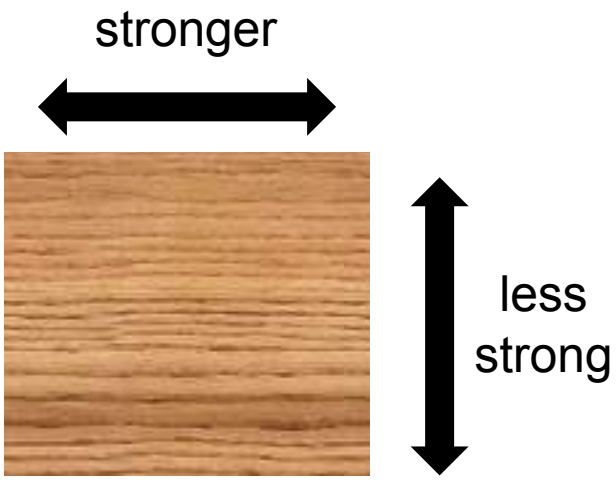
- After support removal



http://electecnik.blogspot.com/2011/12/fds-frame-after-support-removal.html

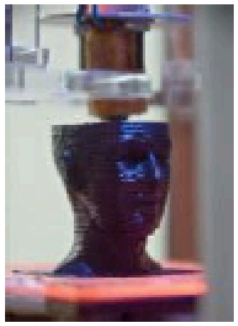
Orientation Determination: Mechanical Properties

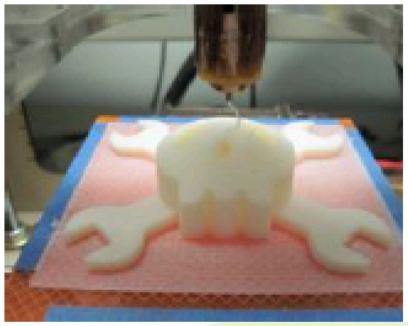
- Orthotropic Materials
  - Mechanical properties are different along each axis
  - Strength/stiffness larger in the direction of the fibers



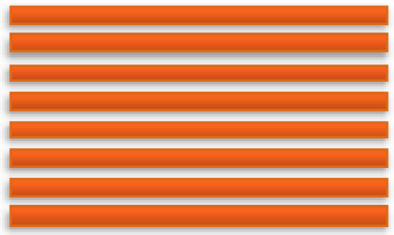
http://en.wikipedia.org/wiki/Orthotropic_material

Orientation Determination: Mechanical Properties

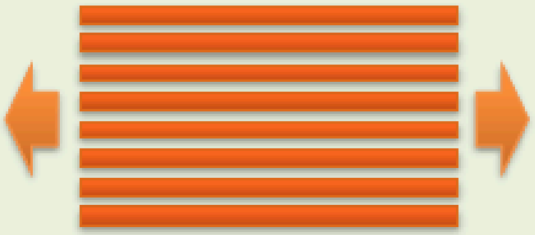




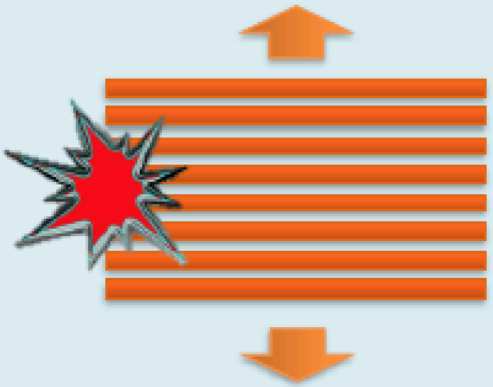
printing direction



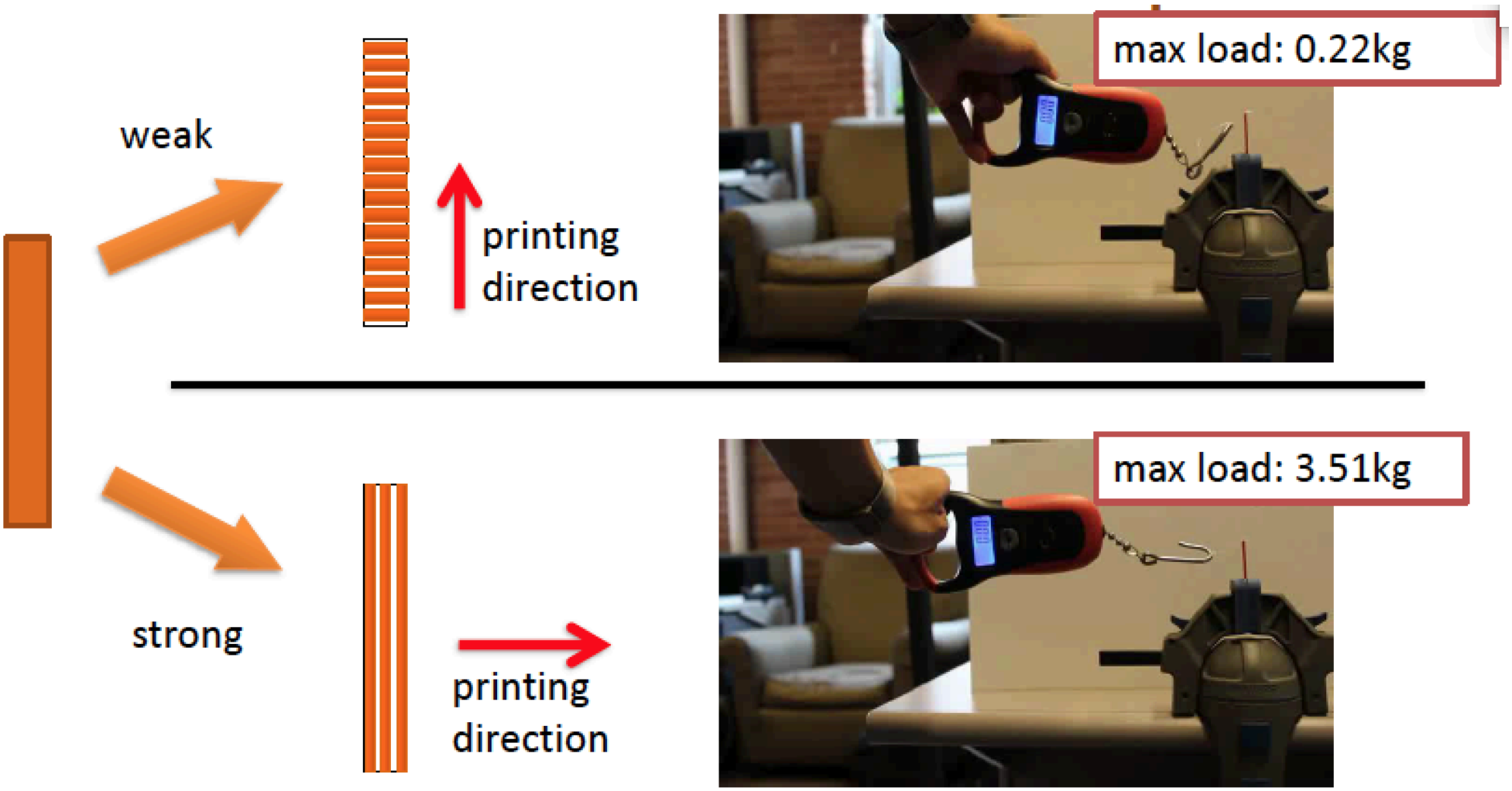
strong



weak

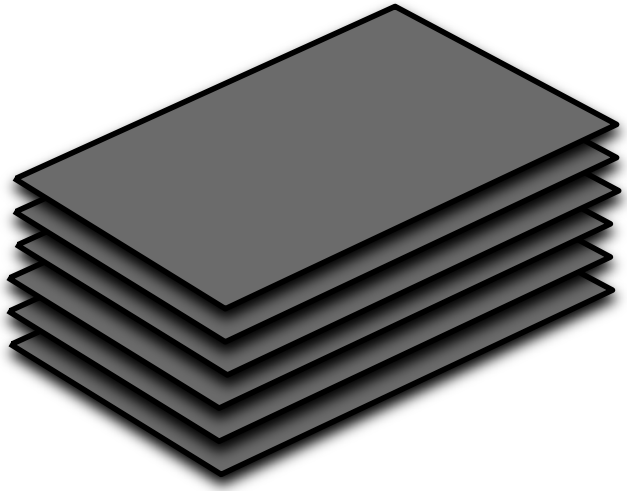


Orientation Determination: Mechanical Properties



Orientation Determination: Mechanical Properties

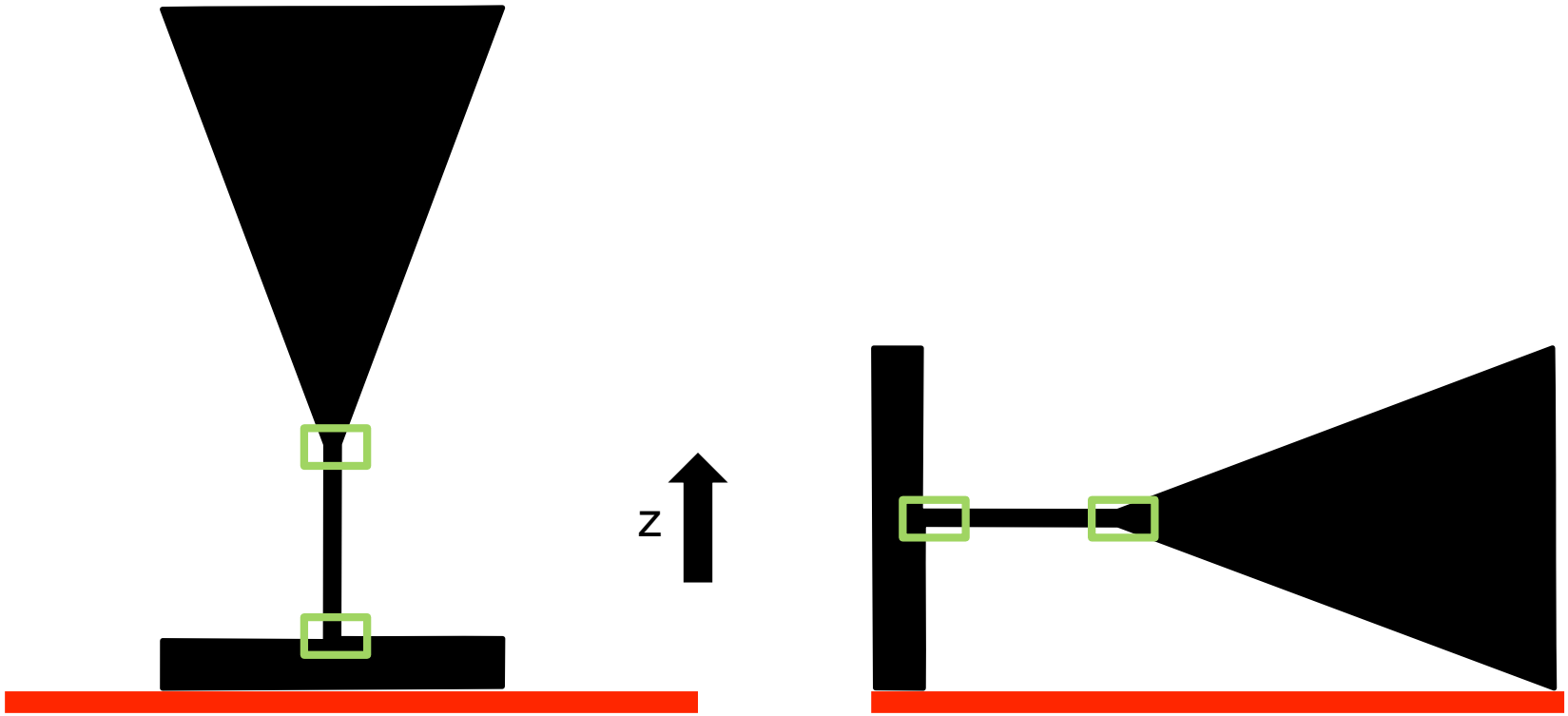
- Transversely isotropic materials
  - Mechanical properties are isotropic within a layer but different across layers



http://en.wikipedia.org/wiki/Transverse_isotropy

Orientation Determination: Mechanical Properties

- Typically strength/stiffness is larger within a layer and smaller across layers



less strong features

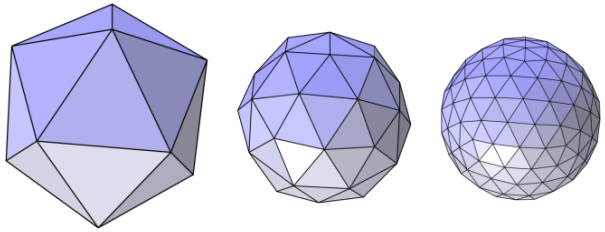
stronger features

Algorithms for Orientation Determination

- Manual placement
  - User is responsible for placing parts on the build tray
- Semi-automated placement
  - User places parts on the build tray
  - System provides feedback on build time, support volume, support contact area, mechanical properties
- Automated placement
  - orientation is computed using optimization according to one or more objectives (build time, support volume, support area, mechanical properties)

Simple Algorithm to Compute Orientation

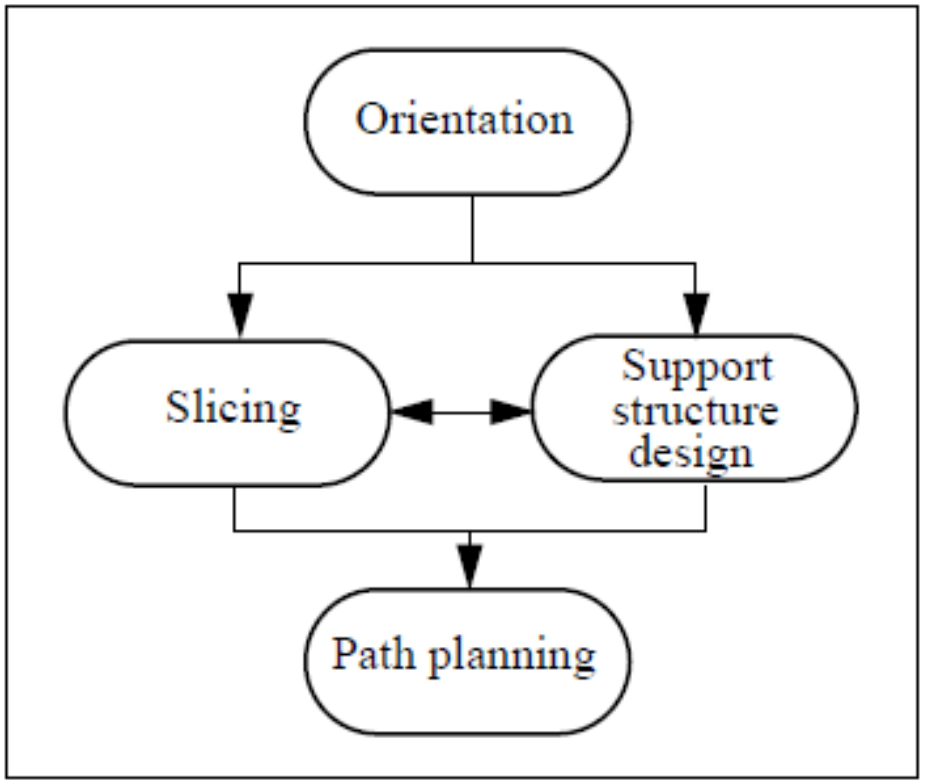
- Exhaustive search
- Compute a uniform set of directions
  - Icosahedron subdivision



- Compute a value of objective function for each direction
  - Build time
  - Support volume
  - Support contact area
  - Mechanical strength
- Pick orientation with the minimum value of the objective
  - Single objective (typical)
  - Multiple objectives need to be weighted (weights are difficult to set)

Process Planning

- Input Model Formats
- Orientation Determination
- Support Structure Determination
- Slicing
- Path Planning
- Machine Instructions

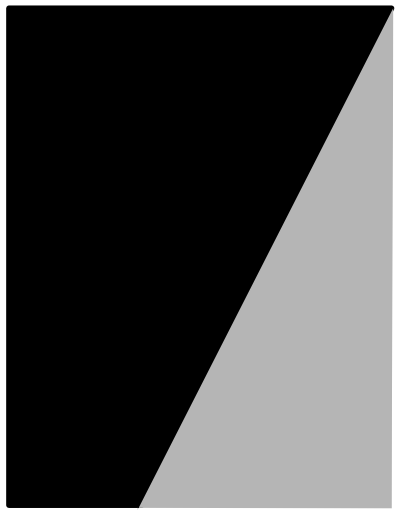


Support Structure Design

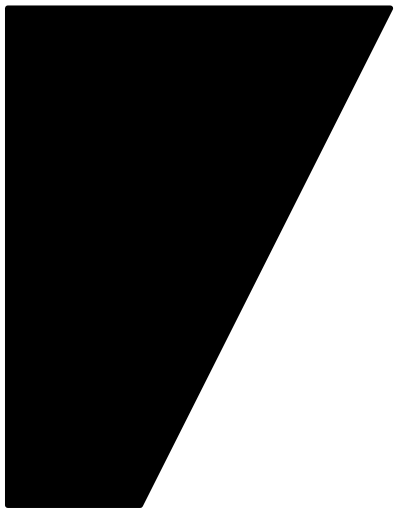
- Do not require support
  - SLS, DMLS, LOM, Plaster-based
- Require support
  - SLA, FDM, phase-change inkjet
- Different goals
  - Prevent curling as the resin hardens
  - Supporting overhangs
  - Maintaining stability (part does not move, tip over)
  - Supporting large flat walls
  - Preventing excessive shrinkage
  - Supporting slanted walls

Support Structure Design

- Based on rules developed from observation
- Depends on a manufacturing method
  - e.g., different rules for phase-change inkjet and FDM (FDM allows surfaces tilted up to 45 degrees



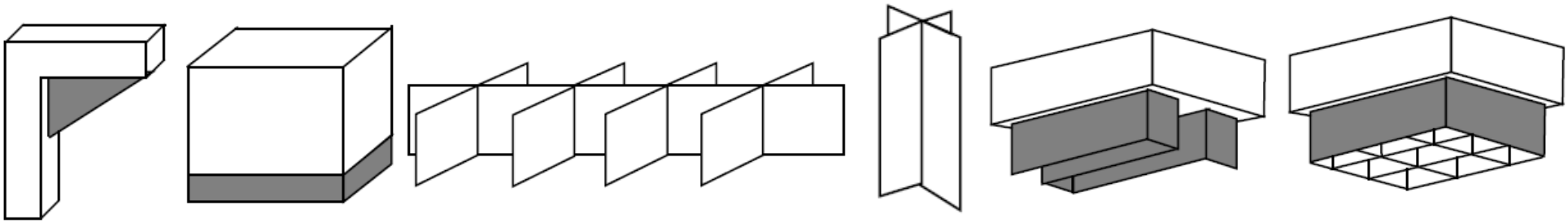
polymer phase change inkjet e.g. Object



FDM

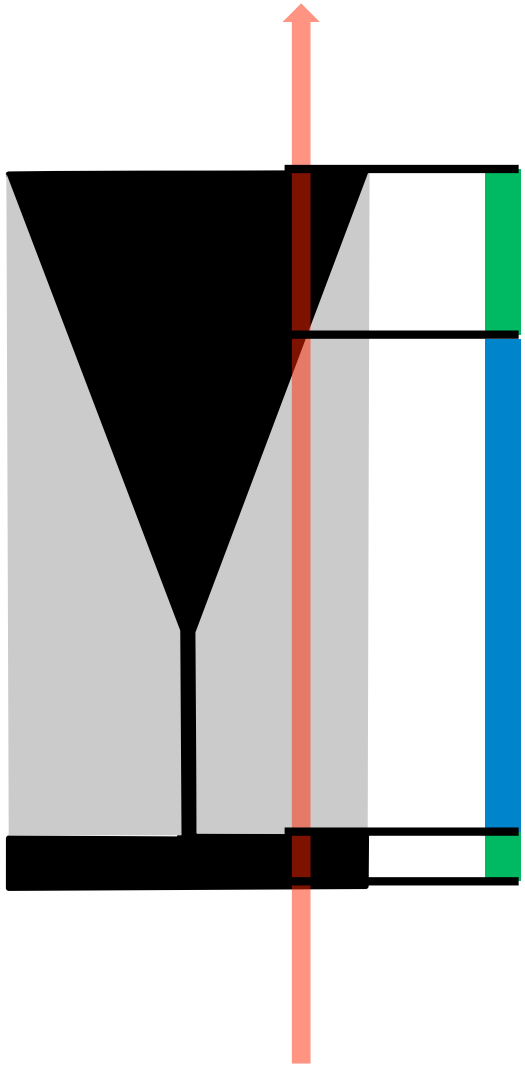
Support Types

- Gusset support
- Base support
- Web support (SLA)
- Column support (SLA)
- Zigzag support (FDM)
- Perimeter and hatch support



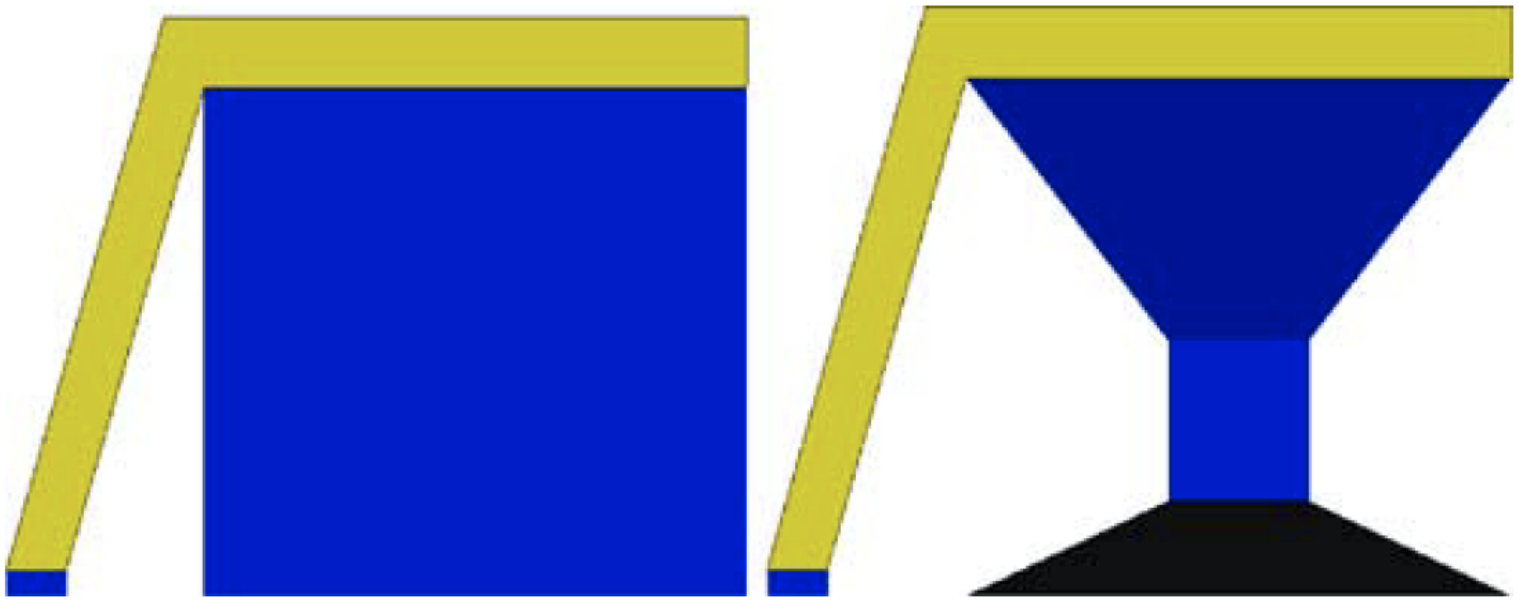
Simple Conservative Algorithm

- Use ray casting in the z direction to compute all intersections for a ray
- Sort intersections in the increasing z to determine intervals inside/ outside of the object
- Any outside intervals before the last inside interval should contain support



Advanced Algorithms

- Minimize the use of support material



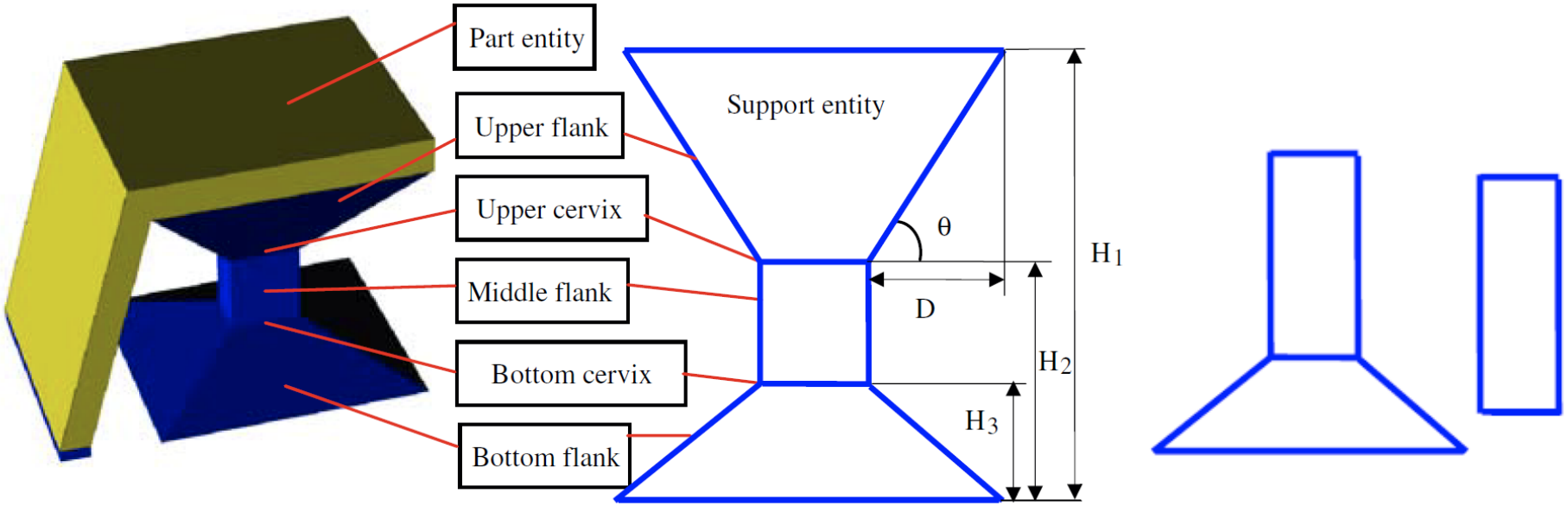
straight wall support

slant wall support

Sloping wall structure support generation for fused deposition modeling. Huang et al. 2009

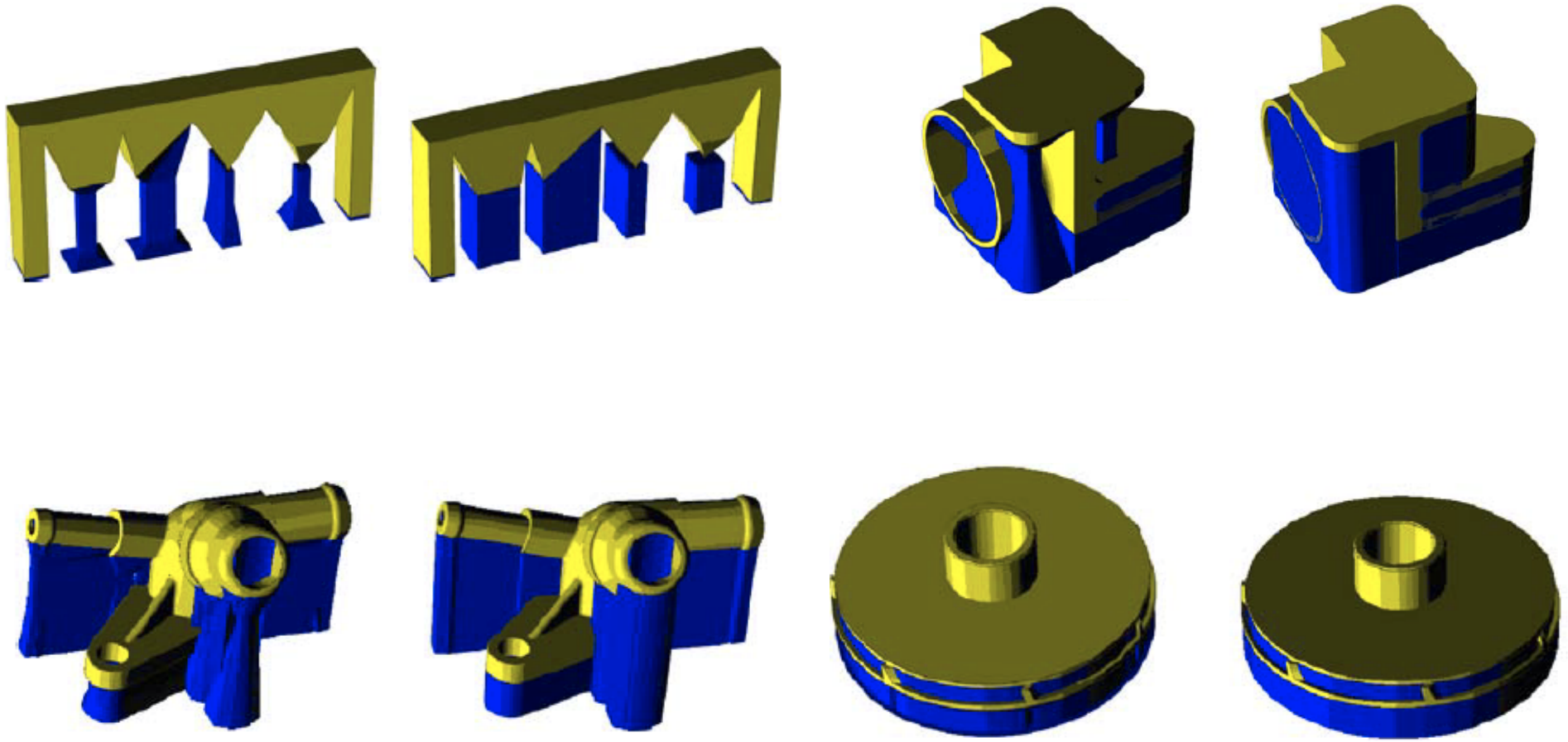
Advanced Algorithms

- Minimize the use of support material



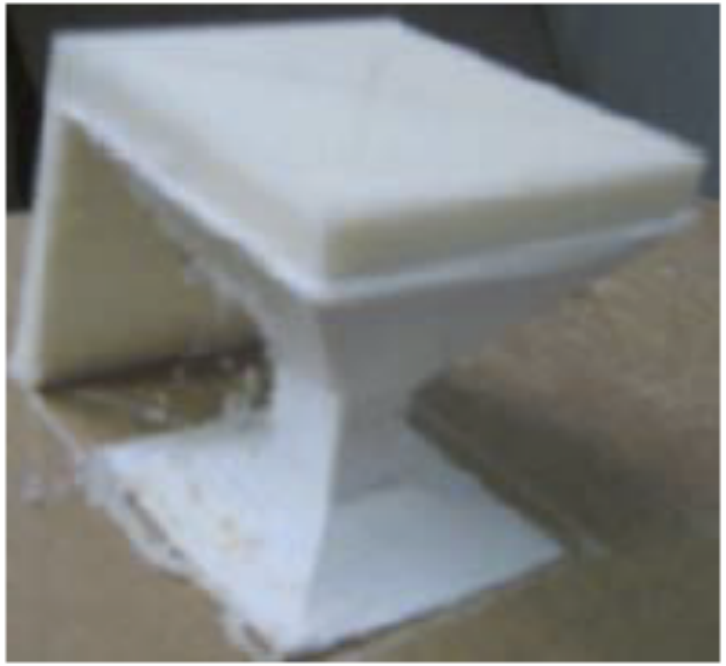
Huang et al. 2009

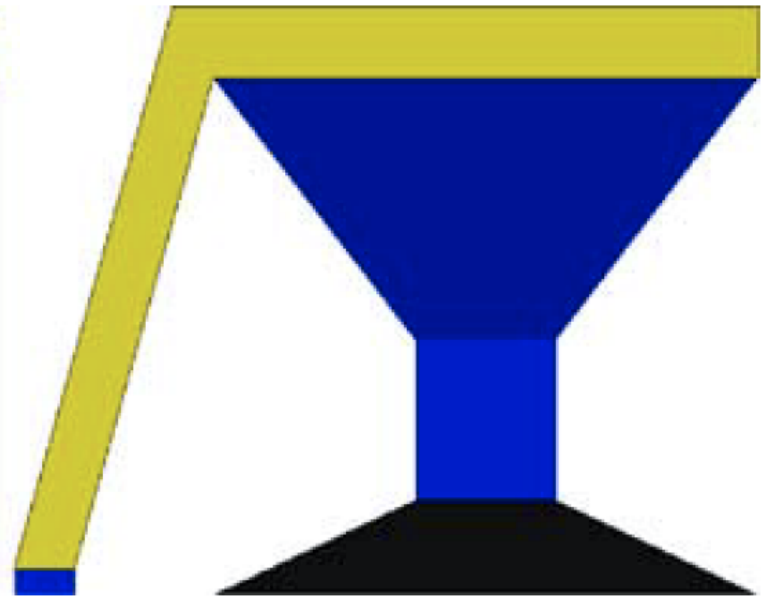
Results: Unoptimized vs. Optimized



Huang et al. 2009

Manufactured Result





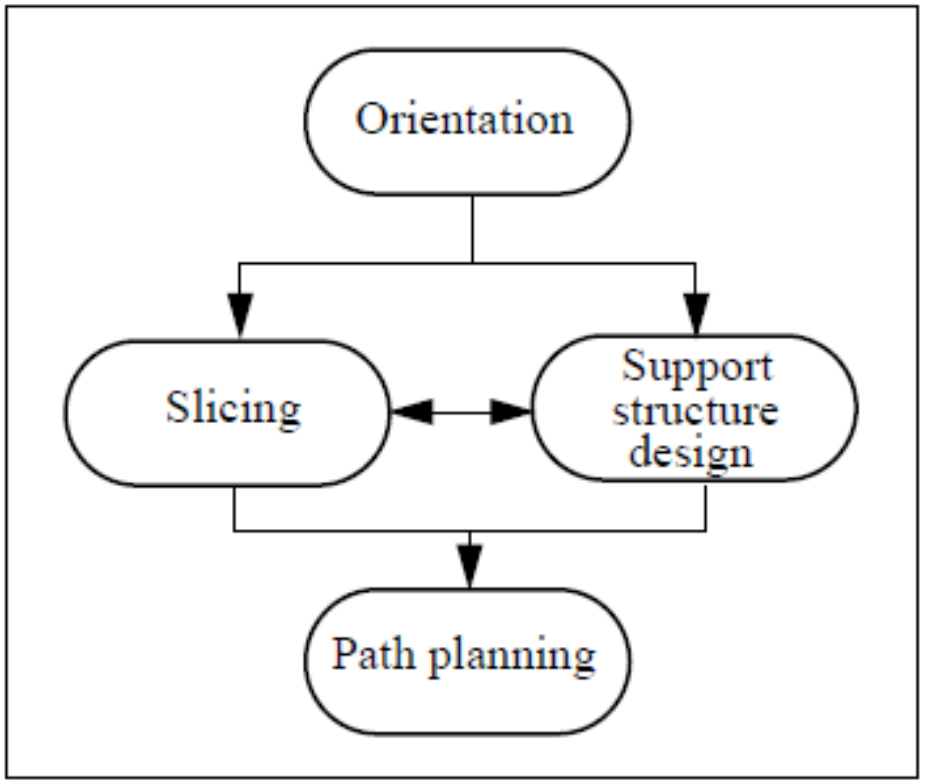
Huang et al. 2009

Advanced Algorithms: Meshmixer

http://www.youtube.com/watch?v=aFTyTV3wwsE

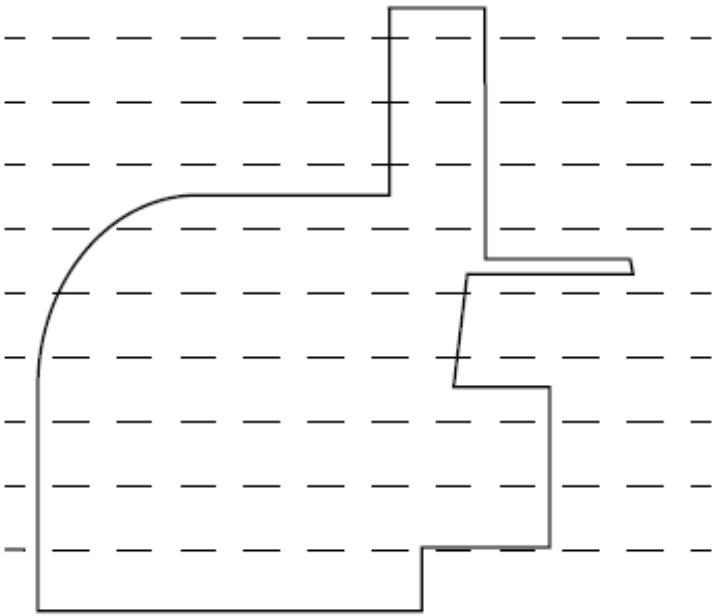
Process Planning

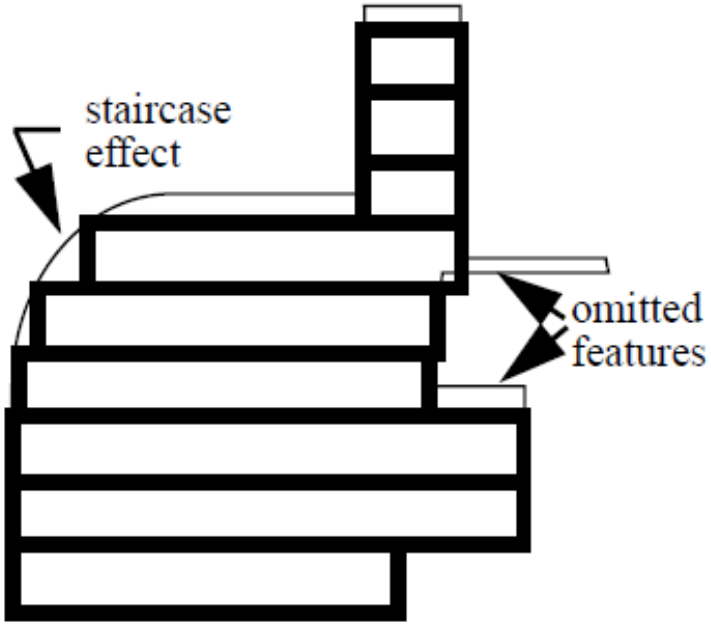
- Input Model Formats
- Orientation Determination
- Support Structure Determination
- Slicing
- Path Planning
- Machine Instructions



Slicing

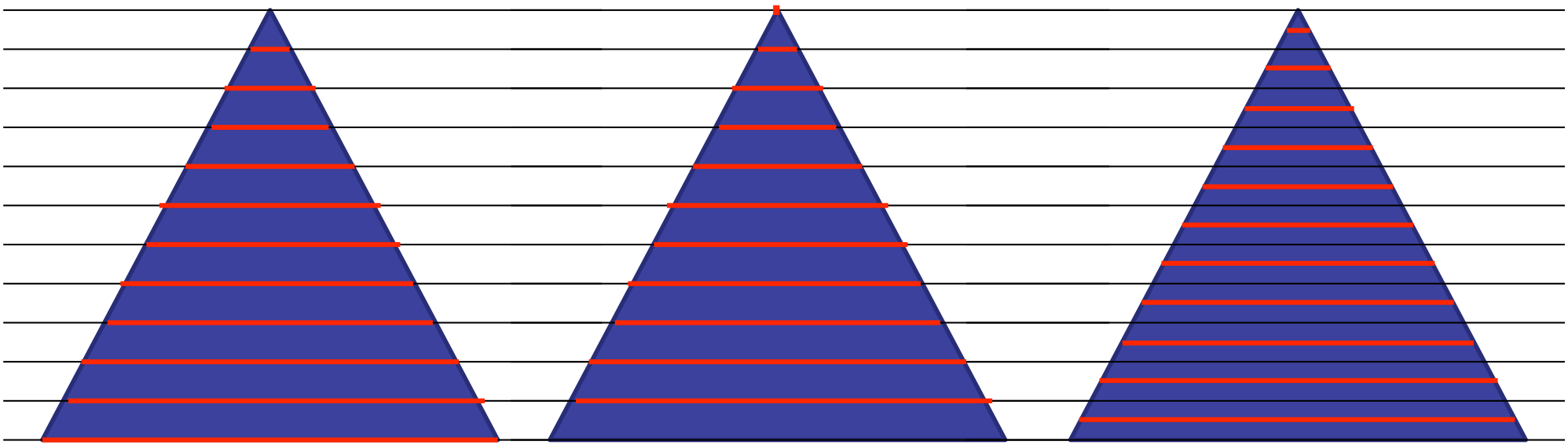
- For a discrete z value, compute an intersection of a plane with a model





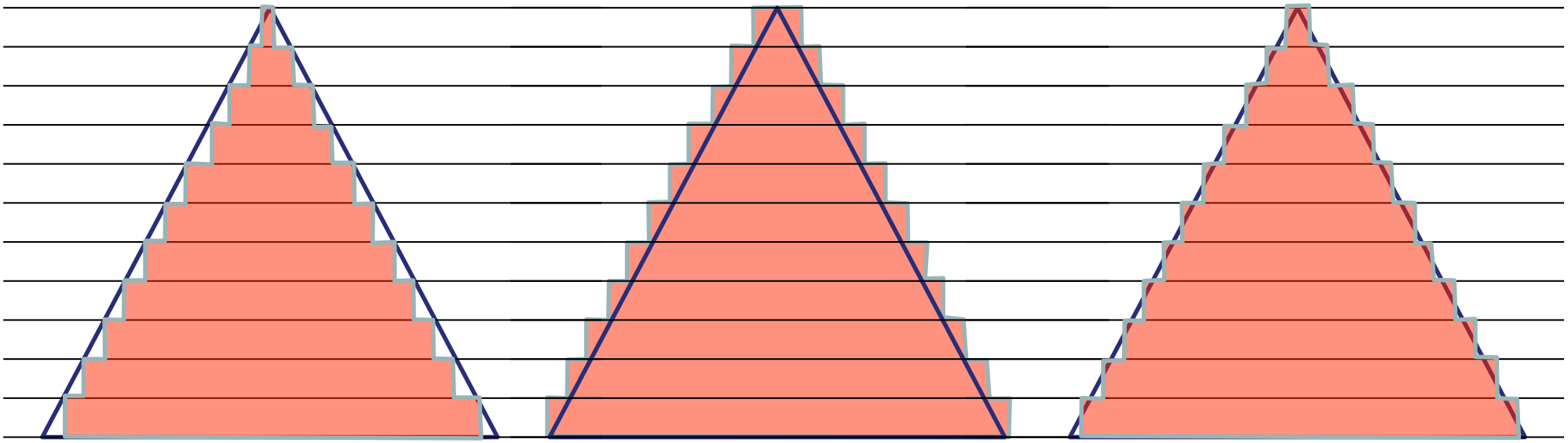
Slicing Issues

- Given that each layer has some finite thickness, which plane to choose?
  - bottom
  - top
  - middle



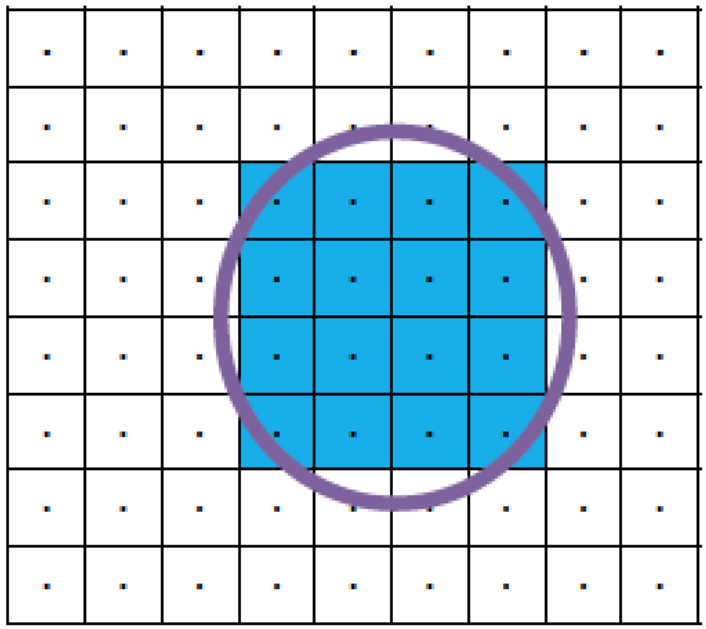
Slicing Issues

- Given that each layer has a finite thickness, which solution to choose?
  - inside the model (negative tolerance/undersize)
  - outside the model (positive tolerance/oversize)
  - best approximation



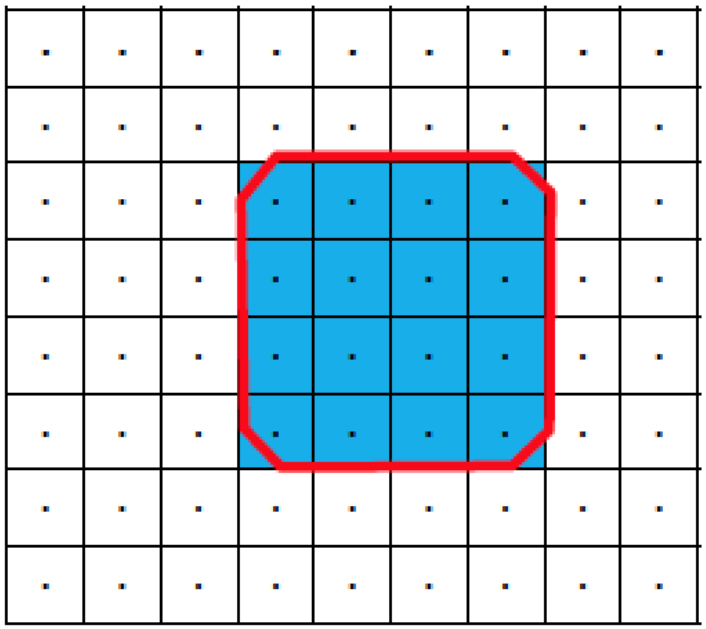
Slicing STL Models: Voxelization

- STL does not store connectivity – “triangle soup”
- Voxelization Algorithm:
  - For each voxel compute inside/outside (Assignment 1)
  - Extract contours



Slicing STL Models: Voxelization

- STL does not store connectivity – “triangle soup”
- Voxelization Algorithm:
  - For each voxel compute inside/outside (Assignment 1)
  - Extract contours

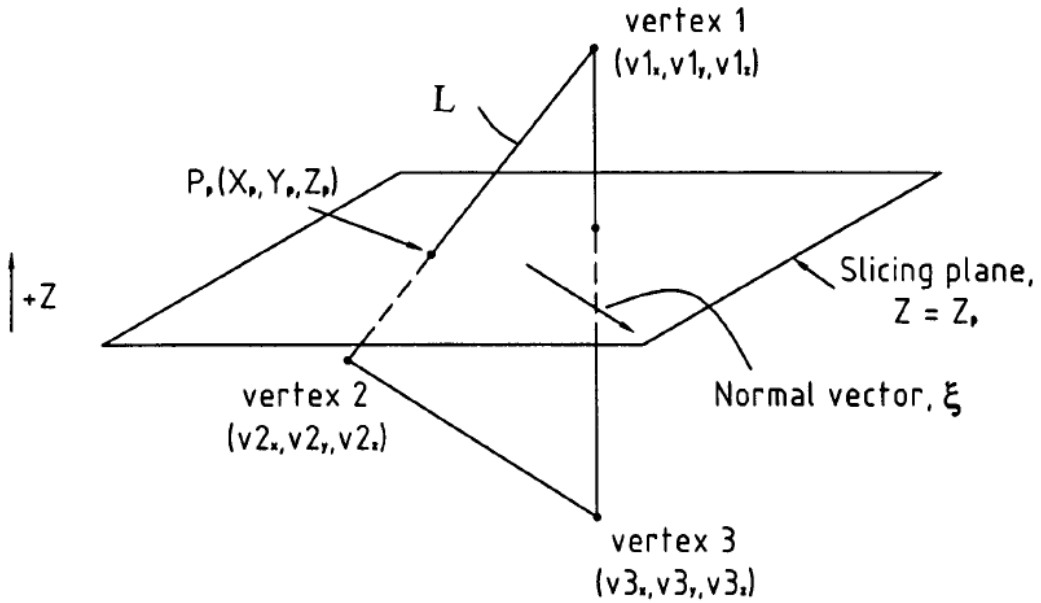


Slicing STL Models

- STL does not store connectivity – “triangle soup”
- Algorithm:
  - For each z plane
    - For each triangle
      - Intersect triangle with the z plane
      - If they intersect, store line segment
    - Connect line segments, store contours

Slicing STL Models

- Algorithm:
  - For each z plane
    - For each triangle
      - Intersect triangle with the z plane
      - If they intersect, store the line segment
    - Connect line segments, store contours

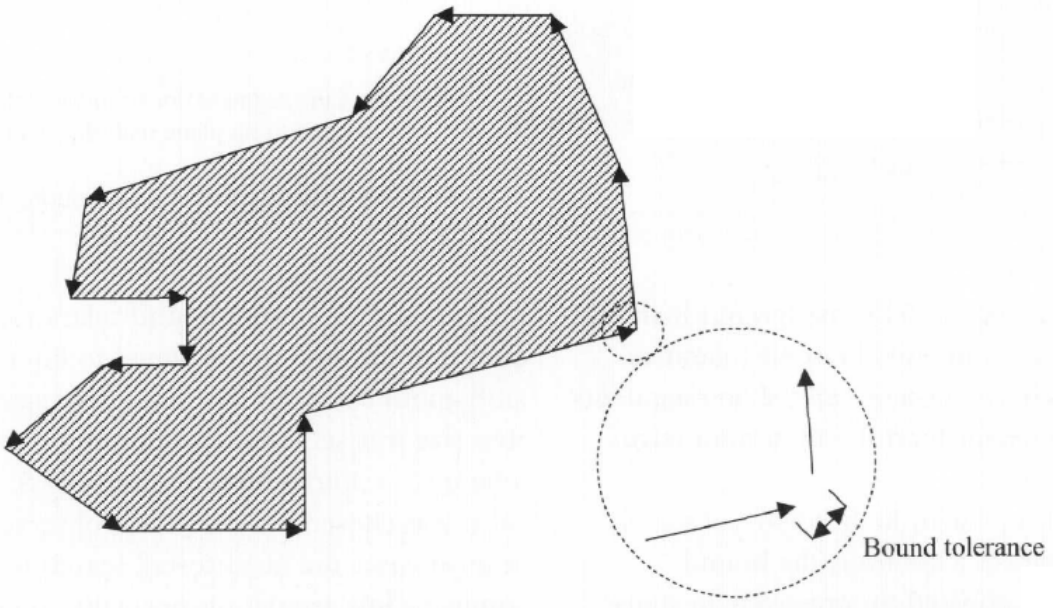


1. **Intersect each edge with the plane**
2. **If two intersection points, connect them to form a line segment**

A tolerant slicing algorithm for layered manufacturing. Choi and Kwok, 2002

Slicing STL Models

- Algorithm:
  - For each z plane
    - For each triangle
      - Intersect triangle with the z plane
      - If they intersect, store the line segment
    - Connect line segments, store contours (need sorting)



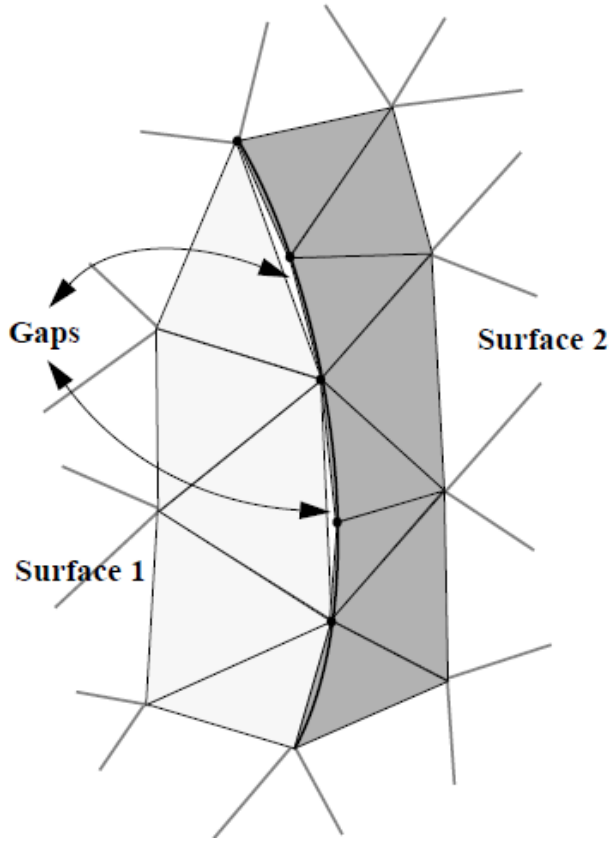
From Choi and Kwok, 2002

Issues
- **spurious line segments**
- **missing line segments**

More epsilons

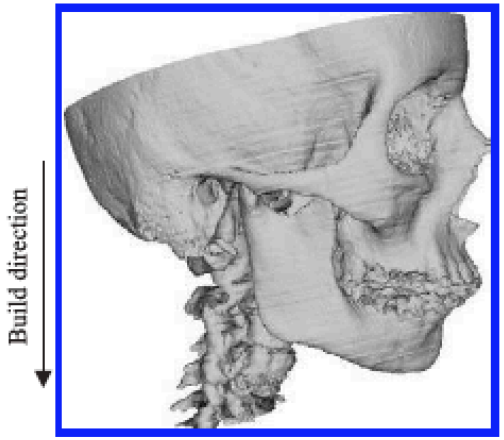
Slicing STL Models

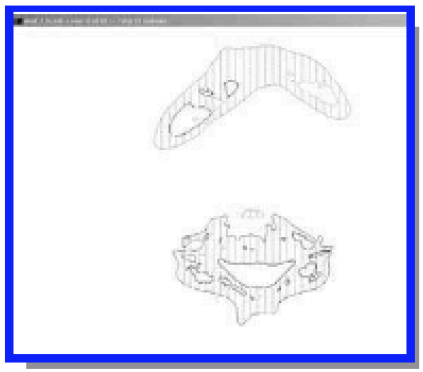
- STL models are not always watertight -> epsilons

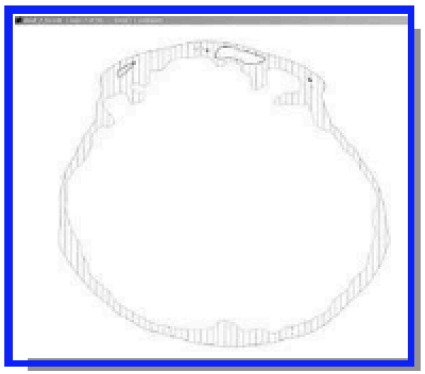


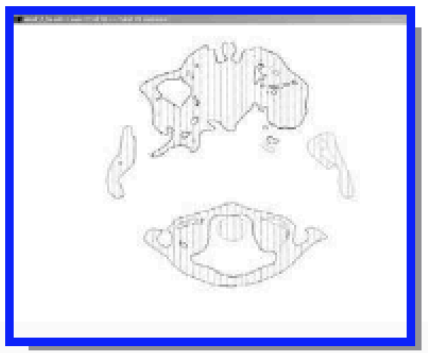
Marsan et al, 1998

Slicing Results



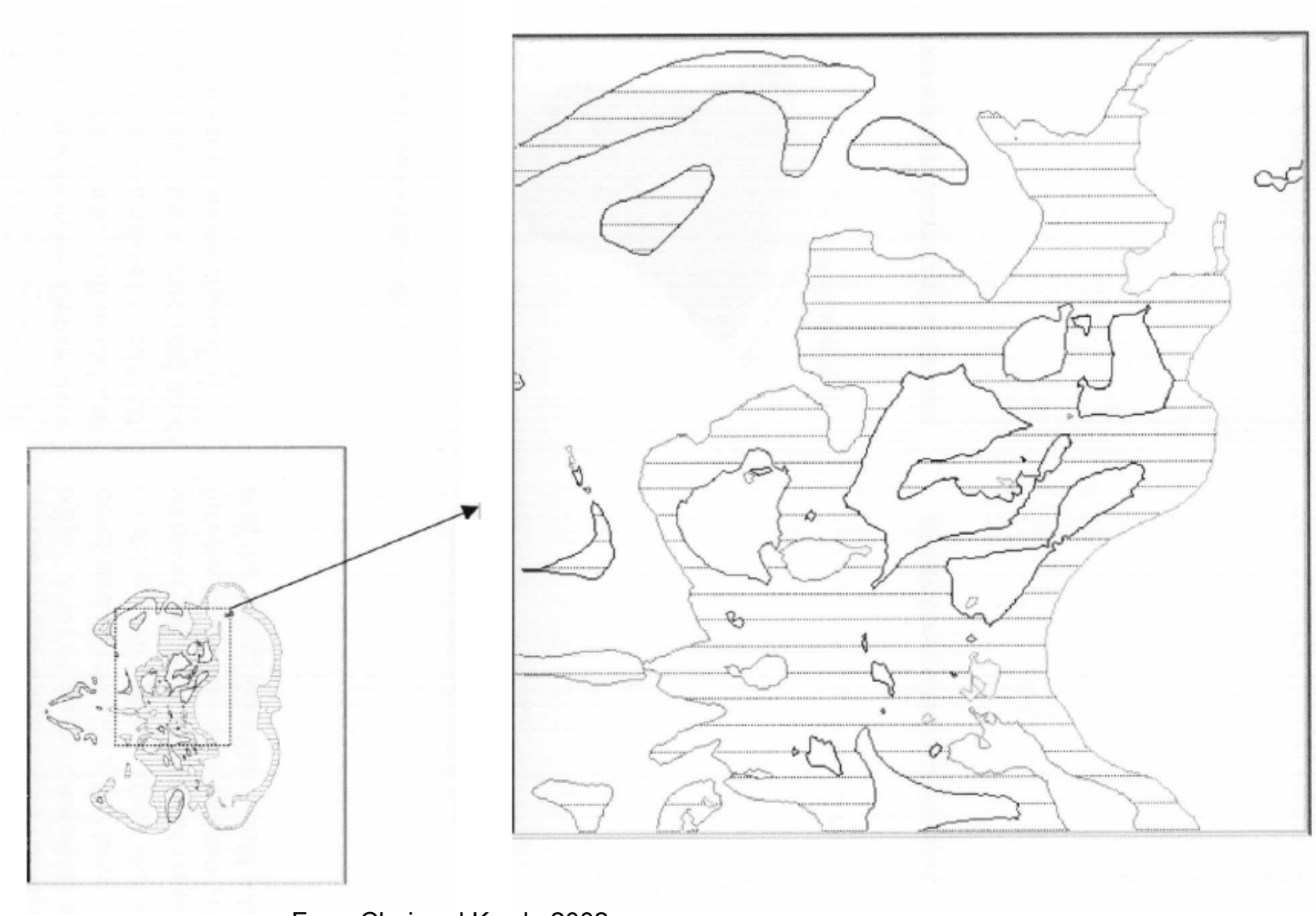






From Choi and Kwok, 2002

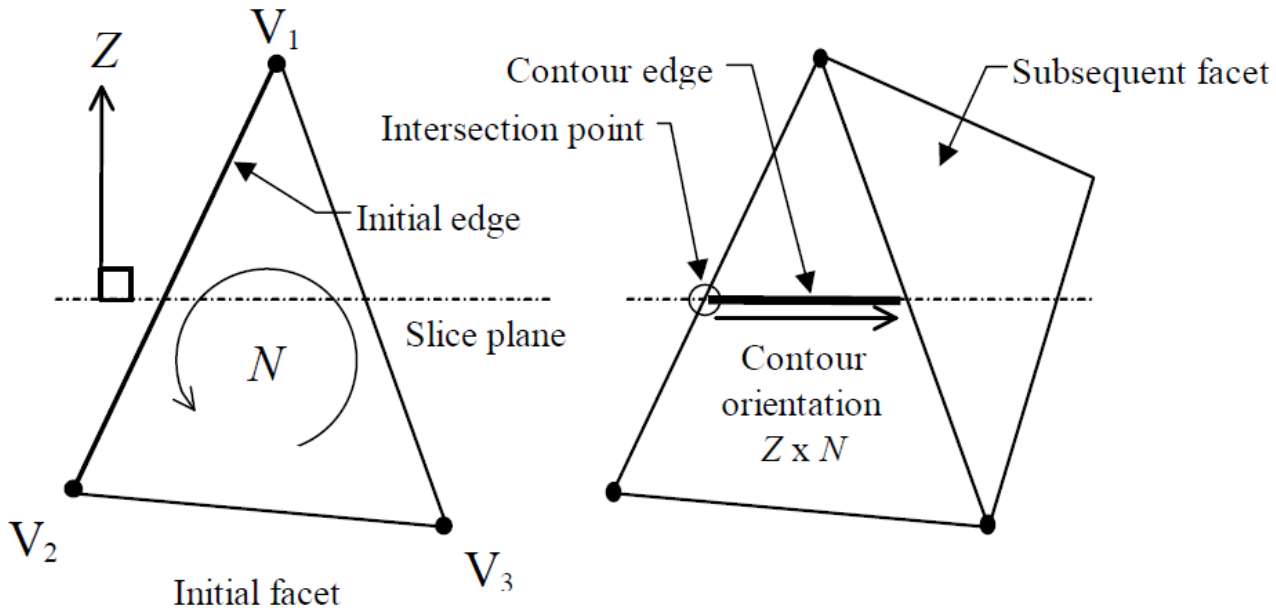
Slicing Results



From Choi and Kwok, 2002

More Efficient Slicing

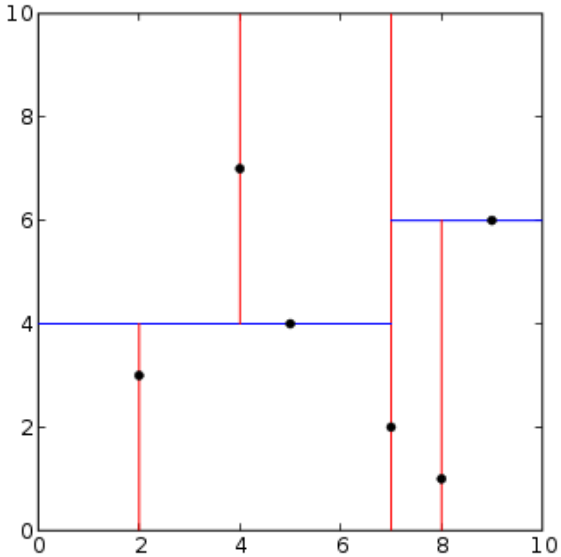
- Precompute topological information (neighboring triangles)
- Find the first intersecting triangle
- Use triangle neighbors to find the next triangle
- **But finding the first intersecting triangle is still slow**



Rock and Wozny [Rock91b]

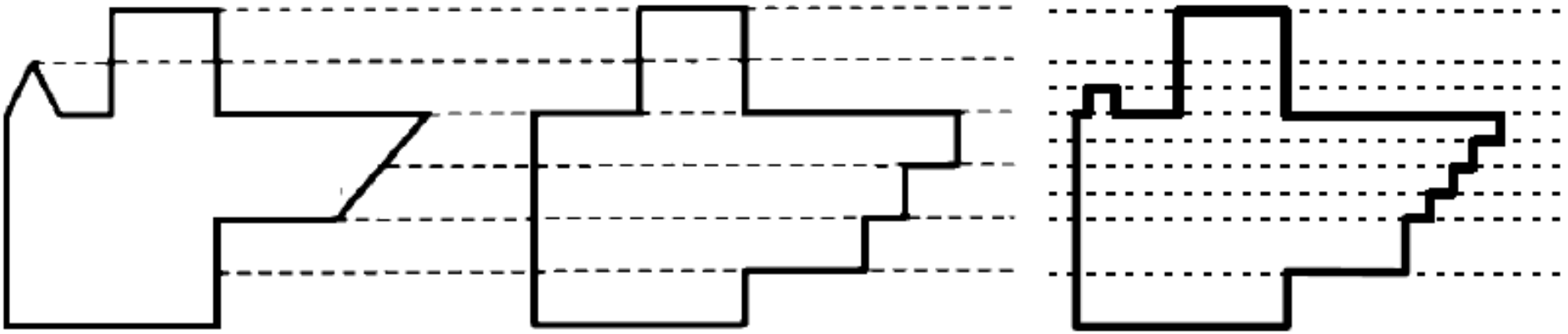
More Efficient Slicing

- Slicing is still a **bottleneck** when working with large models
- Lots of opportunities for very fast algorithms
  - Efficient out-of-core methods
    - when model does not fit in the memory
  - Using computer graphics acceleration data structures
  - Z-sorting
  - GPU-based methods
- Talk to us about project ideas!



Adaptive Slicing

- Slice height is adapted to the input geometry
- Adaptive slicing is rarely used



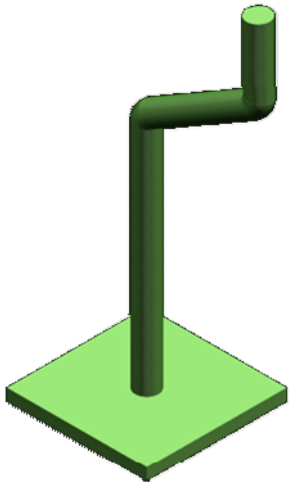
input model

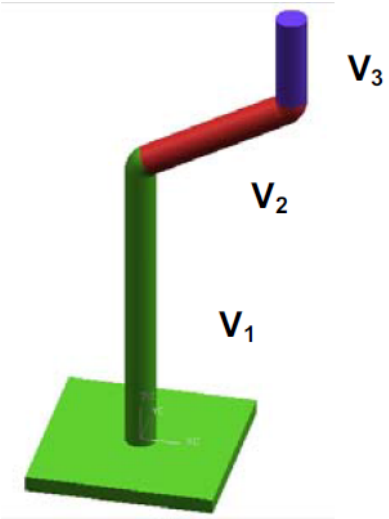
uniform slicing

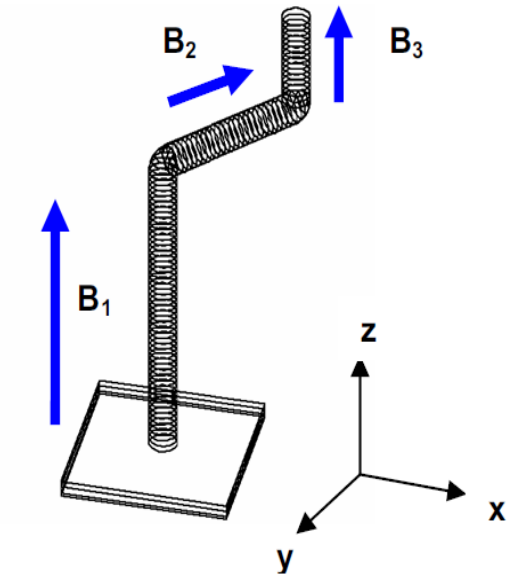
adaptive slicing

Multi-direction Slicing

- Decompose a model into components and manufacturing z directions
- Manufacture each component separately
- Assemble the final model



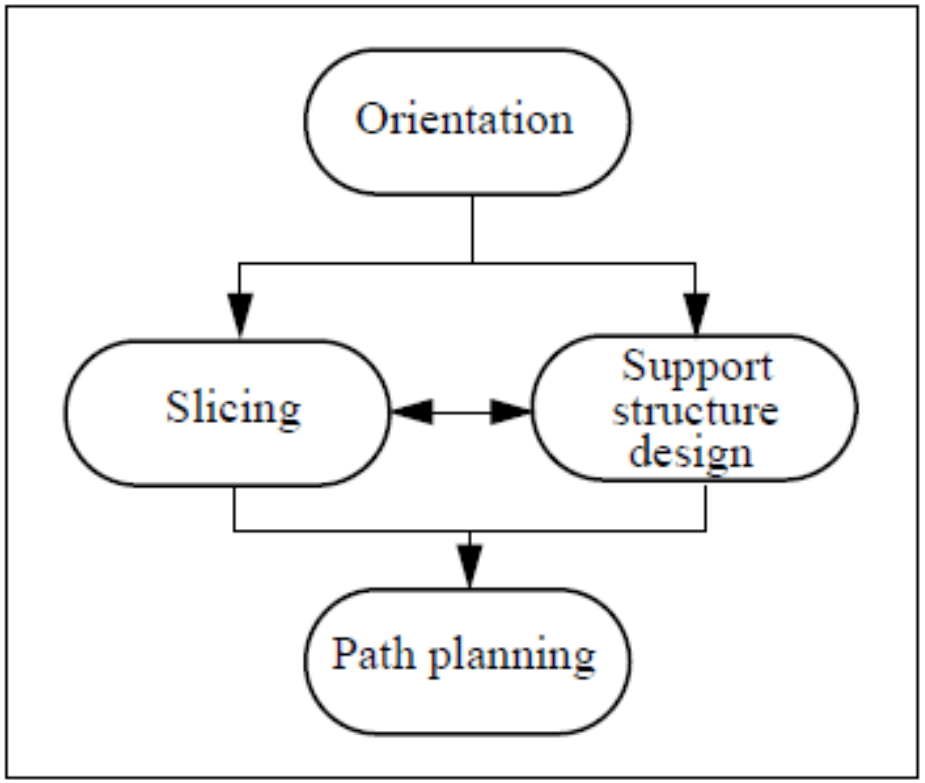




Singh et al. 2003

Process Planning

- Input Model Formats
- Orientation Determination
- Support Structure Determination
- Slicing
- Path Planning
- Machine Instructions



Path Planning

- Two types
  - an entire layer of material is added at once (e.g., LOM)
    - follow the contours of the slice
  - each layer is laid down incrementally (e.g., FDM, SLA)
    - fill the interior (and possibly the contour)

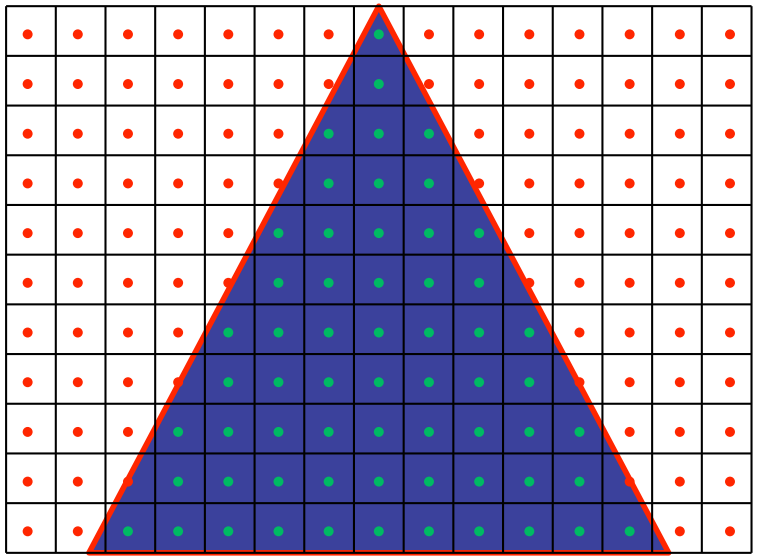
- Paths
  - Affect build time, surface accuracy, stiffness, strength, post-manufacture distortion

Path Planning

- Build time
  - repositioning the tool at the start of a new path
  - accelerating and decelerating for direction changes
- Surface accuracy
  - the filament size
- Distortion
  - materials with a high coefficient of thermal expansion
  - the top layer shrinks when it hardens and it distorts since it is tied to the bottom layer
- Stiffness and strength
  - the area and strength of bonds depends on spacing and the time interval between the tool traversal

Simple Path Planning

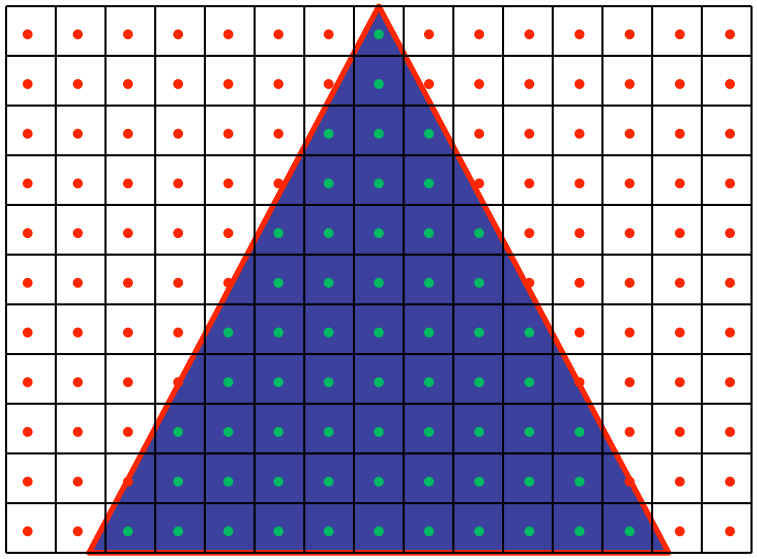
- Superimpose a voxel grid
- Test whether a voxel is inside/outside the model
- Works for DLP 3D printing, plaster-based 3D printing, phase-change inkjets
- Dipping at each position



A project opportunity: principled anti-aliasing algorithms

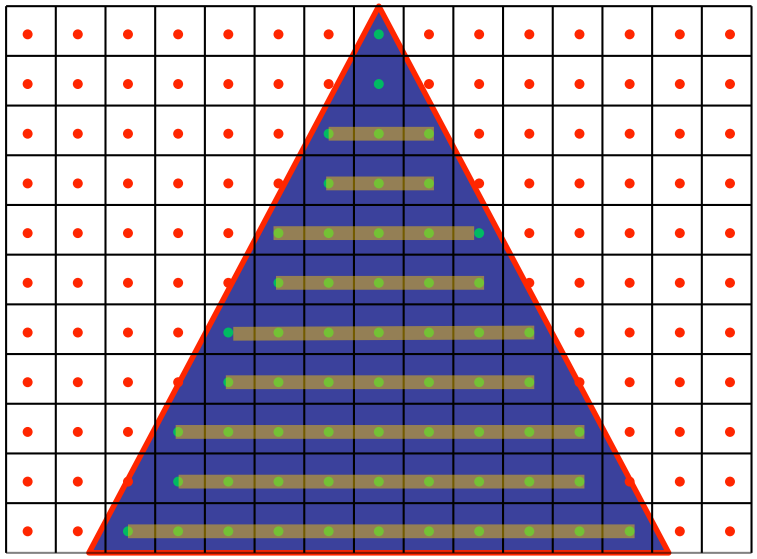
Simple Path Planning

- Superimpose a voxel grid
- Test whether a voxel is inside/outside the model
- Rows or columns are used as tool paths
  - tool starts/stops at transitions between exterior/interior



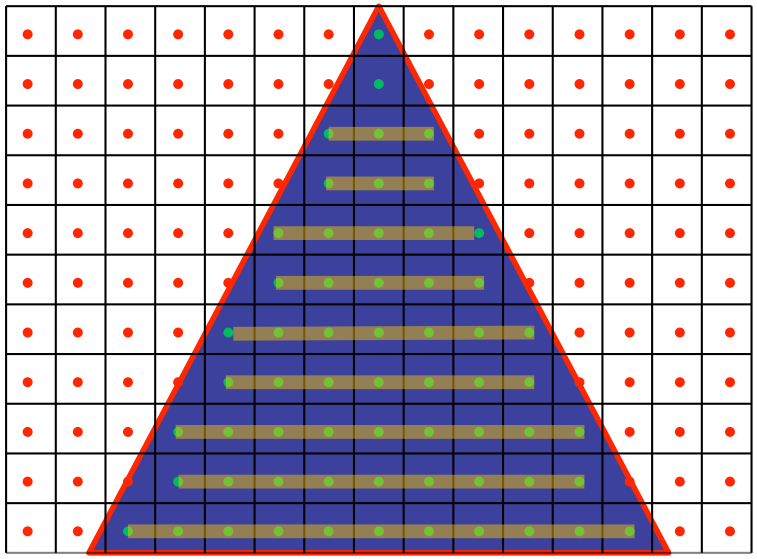
Simple Path Planning

- Superimpose a voxel grid
- Test whether a voxel is inside/outside the model
- Rows or columns are used as tool paths
  - tool starts/stops at transitions between exterior/interior



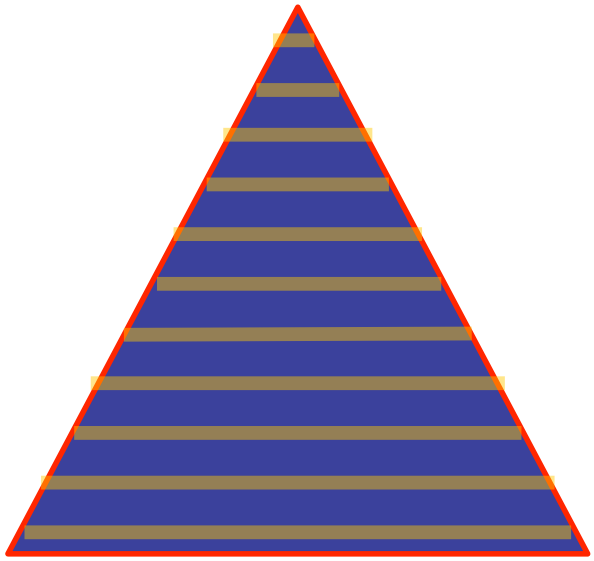
Simple Path Planning

- Superimpose a voxel grid
- Test whether a voxel is inside/outside the model
- Rows or columns are used as tool paths
  - tool starts/stops at transitions between exterior/interior



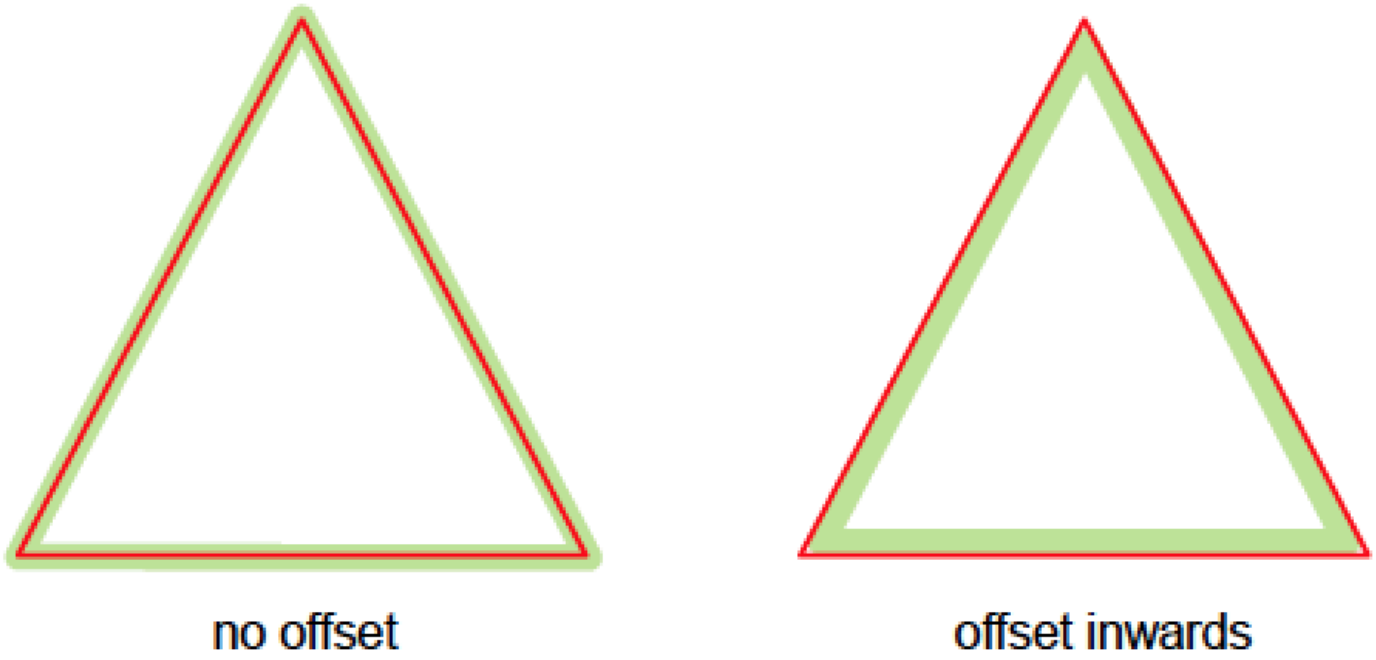
Simple Path Planning

- Cast parallel uniformly spaced rays on the slicing plane
- Compute intersection intervals with the model
- The tool is turned on/off at interval intersections



Tracing Contour

- Improve accuracy of the surface
- Optional: offset inwards by distance equal to the filament radius

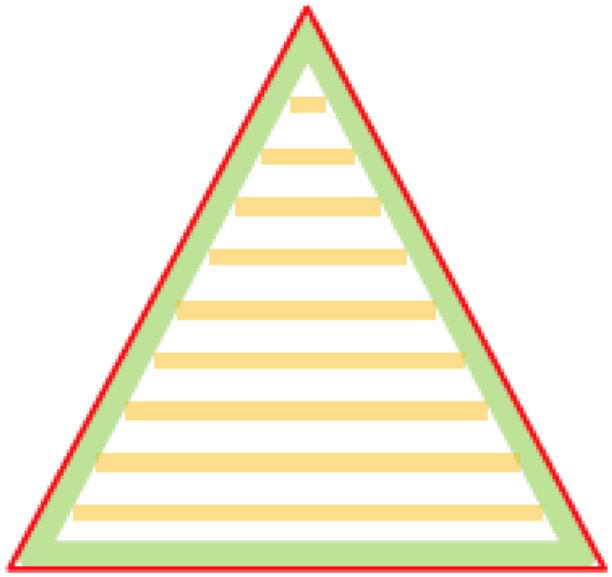


Tracing Contour

- Allows manufacturing hollow objects, some overhangs, some tilted surfaces
- Reduces frequency of tool repositioning
- Reduces support structures

Tracing Contour

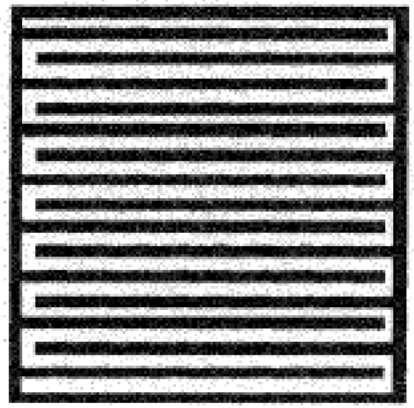
- Can be combined with filling the interior
- Interior fill paths do not extend from border to border
  - stopped short of the contour

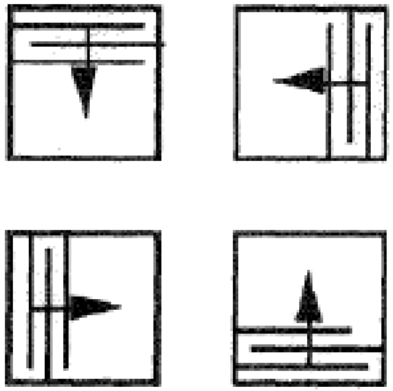


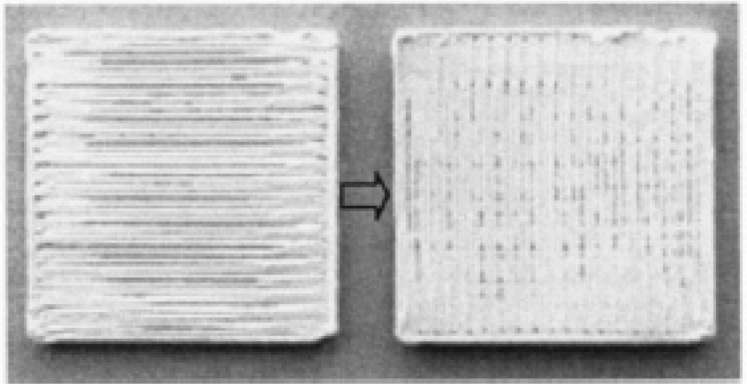
contour offset inwards + interior fill

Tracing Contour

- The interior can be completely filled

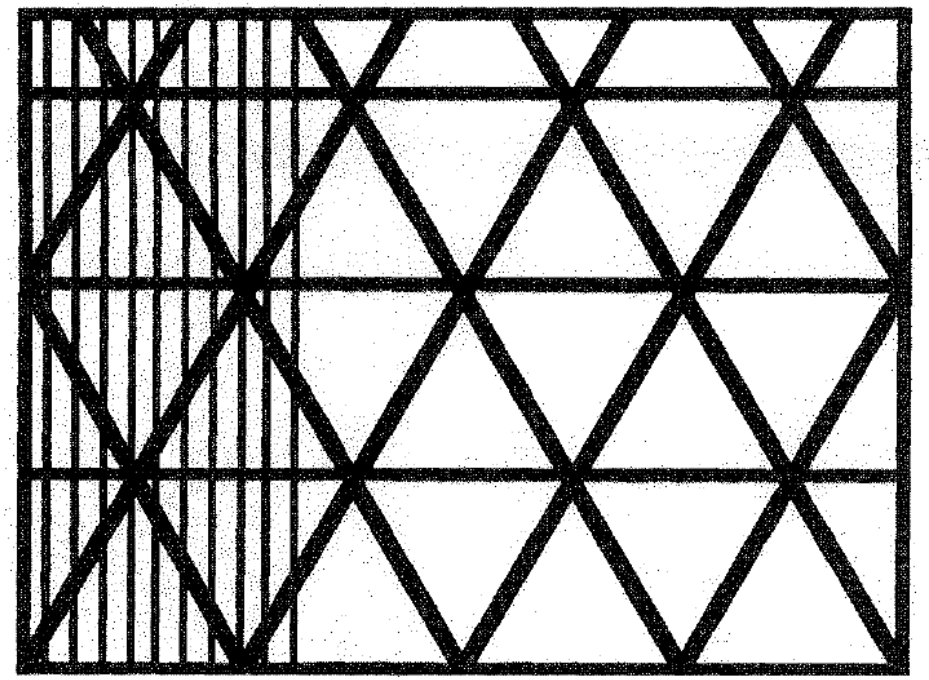






Advanced Fill Patterns

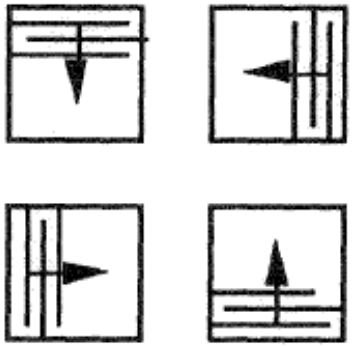
- TriHatch (developed by 3D Systems)
  - developed for stereolithography (SLA)
  - interior layer is filled with equilateral triangles
  - skins fills on the bottom and top of the part



Horton et al 1993

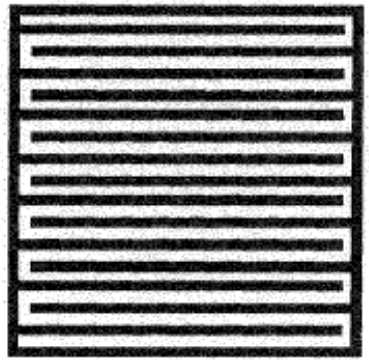
Advanced Fill Patterns

- STARWEAVE (developed by 3D Systems)
  - Scan direction in each layer is perpendicular to the previous layer
  - Alternate layers are staggered (shifted by ½ filament)
  - Fill paths do not extend from border to border



alternate sequencing

staggered weave

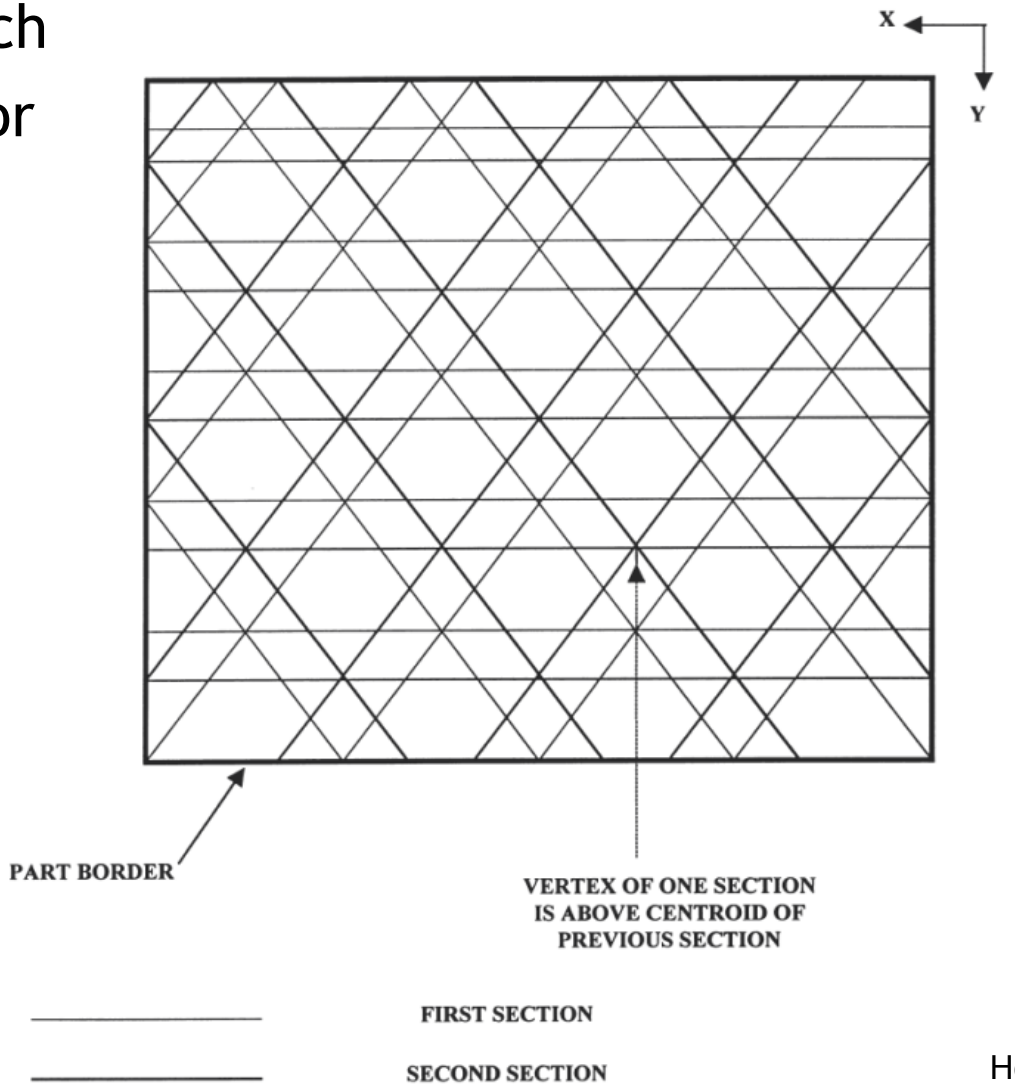


retracted hatch

Horton et al 1993

More Fill Patterns

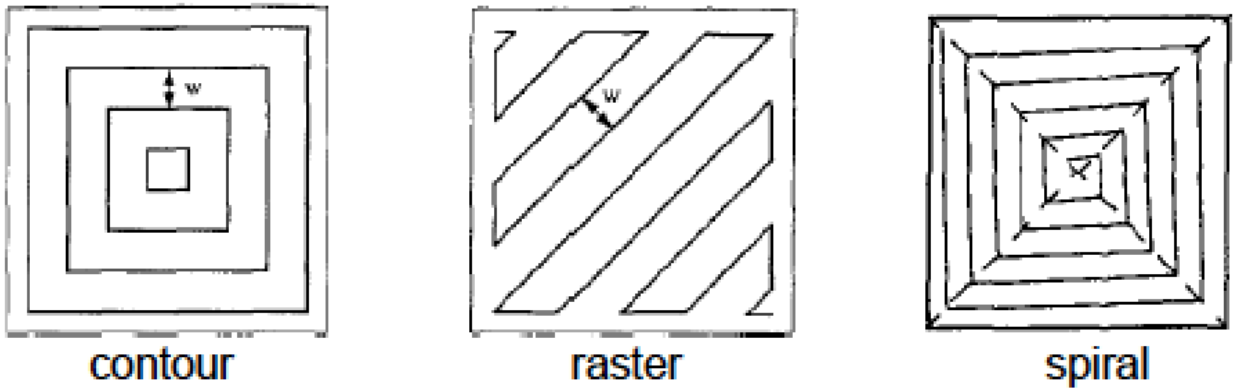
- QuickCast
  - Similar to TriHatch
  - Patterns offset for each layer

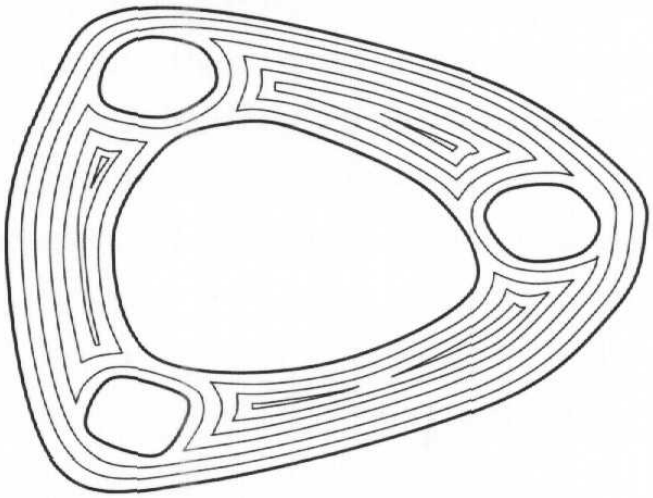


Horton et al 1993

More Fill Patterns

- Criteria: print speed, structural properties, weight vs strength, etc

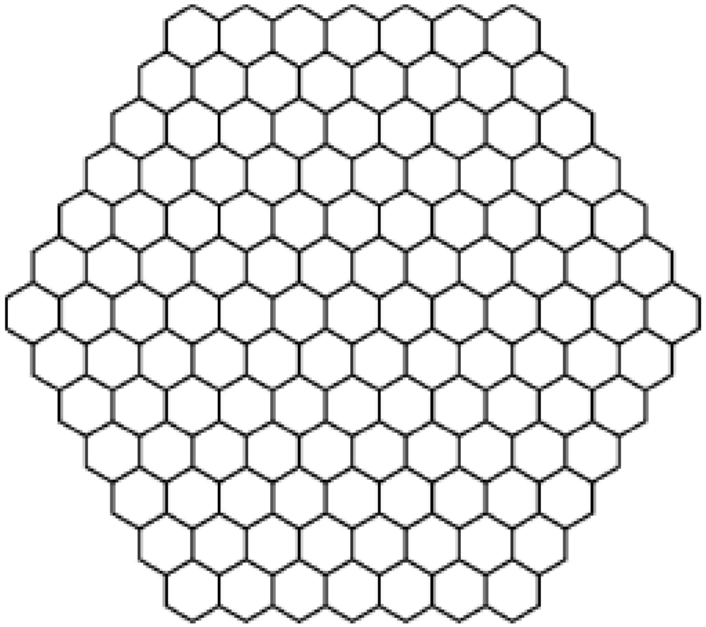


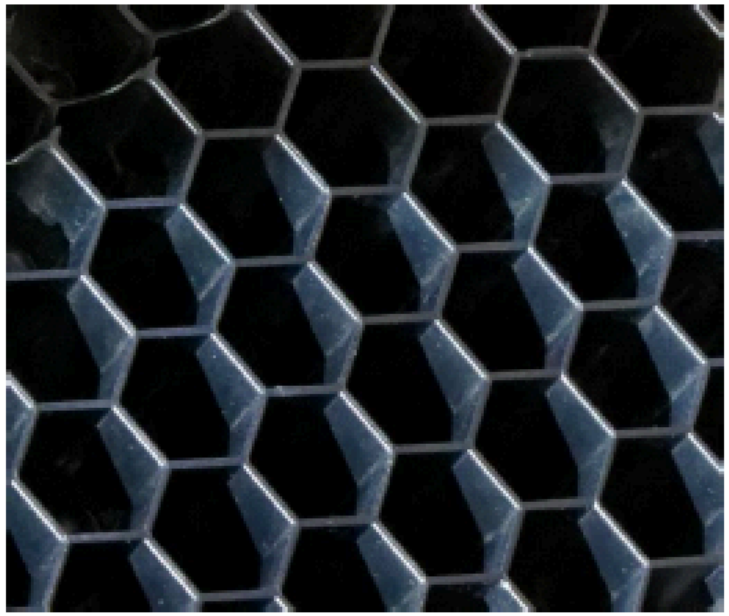


A project opportunity

More Fill Patterns

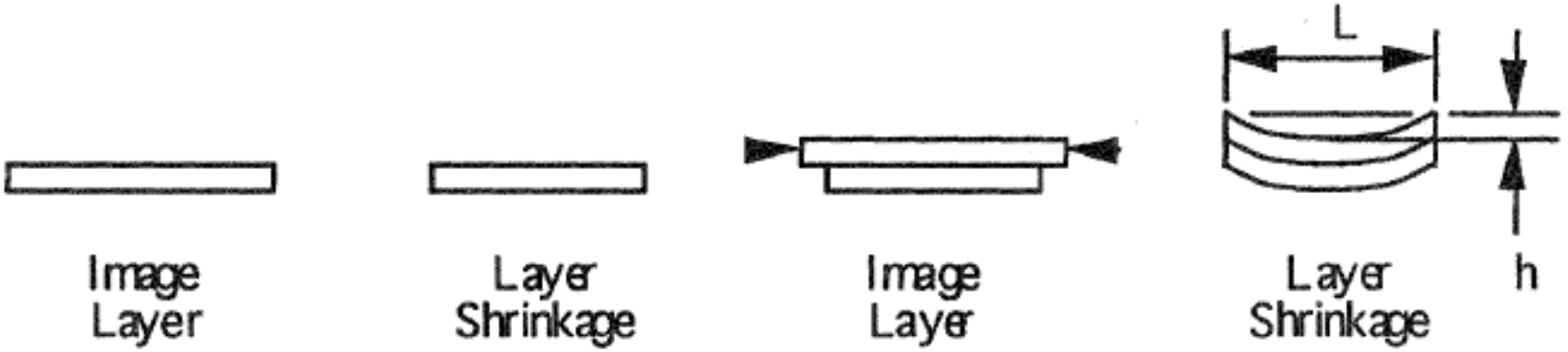
- A ***honeycomb-cell structure*** is a good trade-off between overall weight and strength

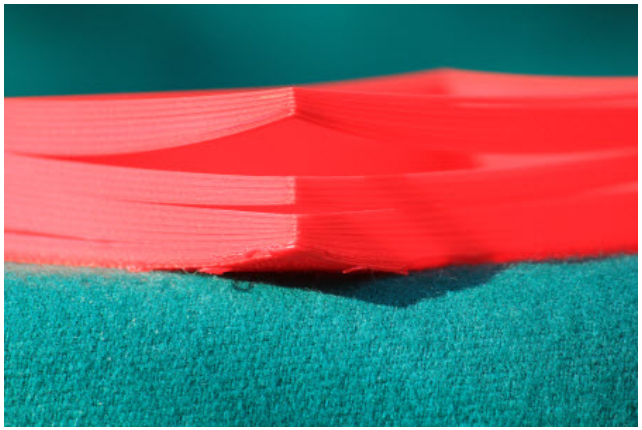


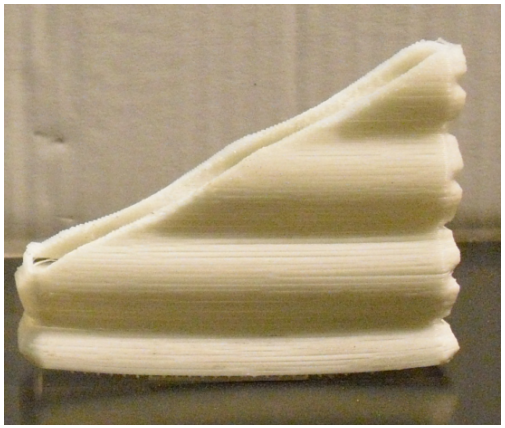


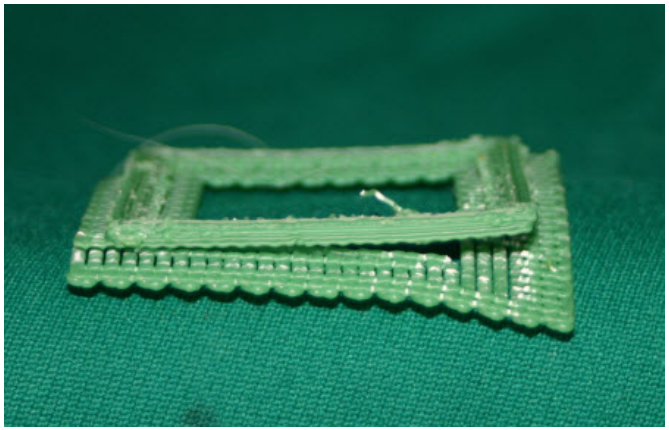
Material Shrinkage/Warping

- Materials can shrink when cooling down/curing
- Path patterns can minimize this effect









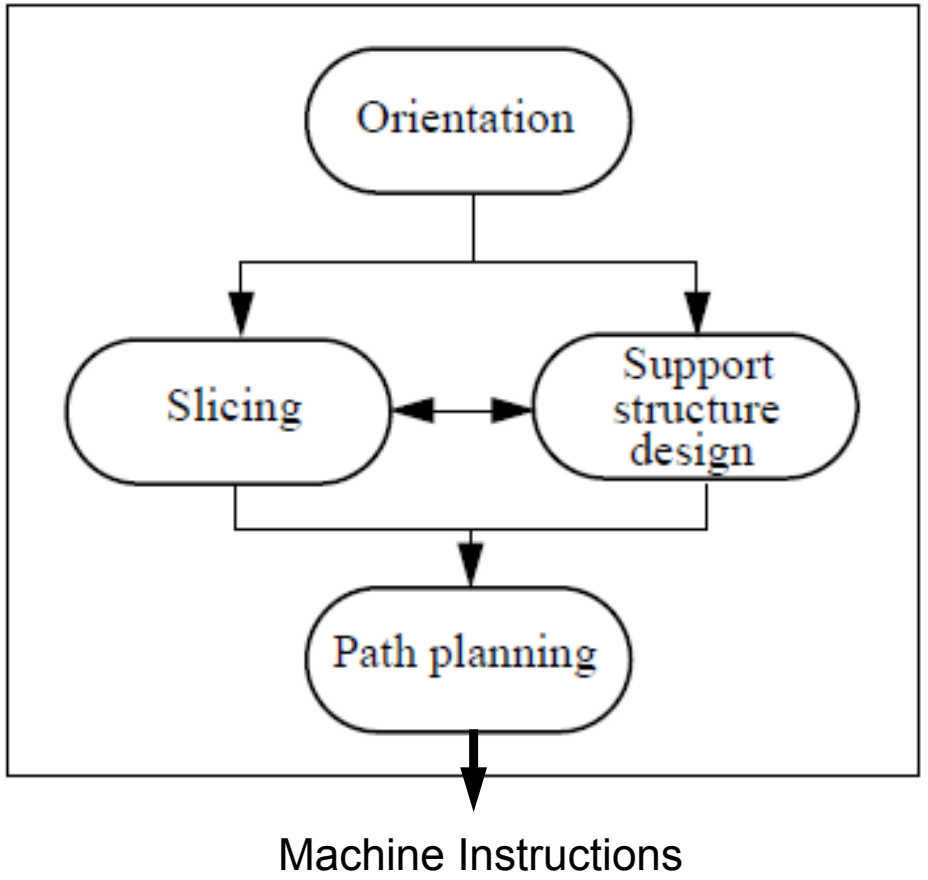
Source: http://cc-products.de, http://www.deelip.com

Path Planning Summary

- Minimal input is the slice description
- Optimal scan strategy also requires characteristics of the particular process
  - Both geometric and process data are important
  - Structural properties of the object must be specified

Process Planning

- Input Model Formats
- Orientation Determination
- Support Structure Determination
- Slicing
- Path Planning
- Machine Instructions



Machine Instructions

- G-Code
  - machine-specific 2D format for the commands that control the laser beam

G-code

- Numerical control (NC) programming language
- Developed at MIT in 1950s
- Used for CNC milling machines, now for many 3D printers
- Sample Instructions
  - **G00: Rapid move**
    - does not necessarily move in a single straight line between start point and end point. It moves each axis at its max speed until its vector is achieved.
  - **G01: Linear interpolation**
    - specify the start and end points, and the control automatically calculates the intermediate points to pass through that will yield a straight line
  - **G02: Circular interpolation, clockwise**

G-code Example

```
G17 G20 G90 G94 G54
G0 Z0.25
X-0.5 Y0.
Z0.1
G01 Z0. F5.
G02 X0. Y0.5 I0.5 J0. F2.5
X0.5 Y0. I0. J-0.5
X0. Y-0.5 I-0.5 J0.
X-0.5 Y0. I0. J0.5
G01 Z0.1 F5.
G00 X0. Y0. Z0.25
```

This program draws a 1" diameter circle about the origin in the X-Y plane.

seek the Z-axis to 0.25“
travel to X=-0.5 and Y=0.0

lower back to Z=0.0.
draw a clockwise circle at a slow feed rate.

lift the Z-axis up 0.1“
seek back to X=0.0, Y=0.0, and Z=0.25

https://github.com/grbl/grbl/wiki/G-Code-Examples

That’s All for Today!