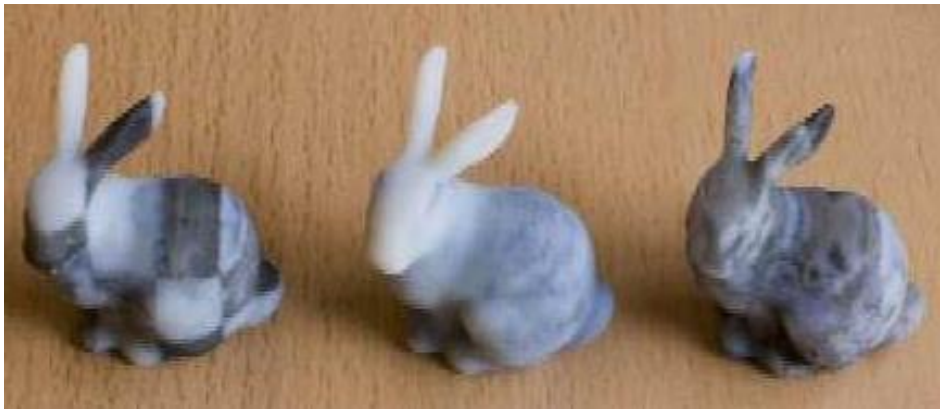
3D Printing Software Pipeline

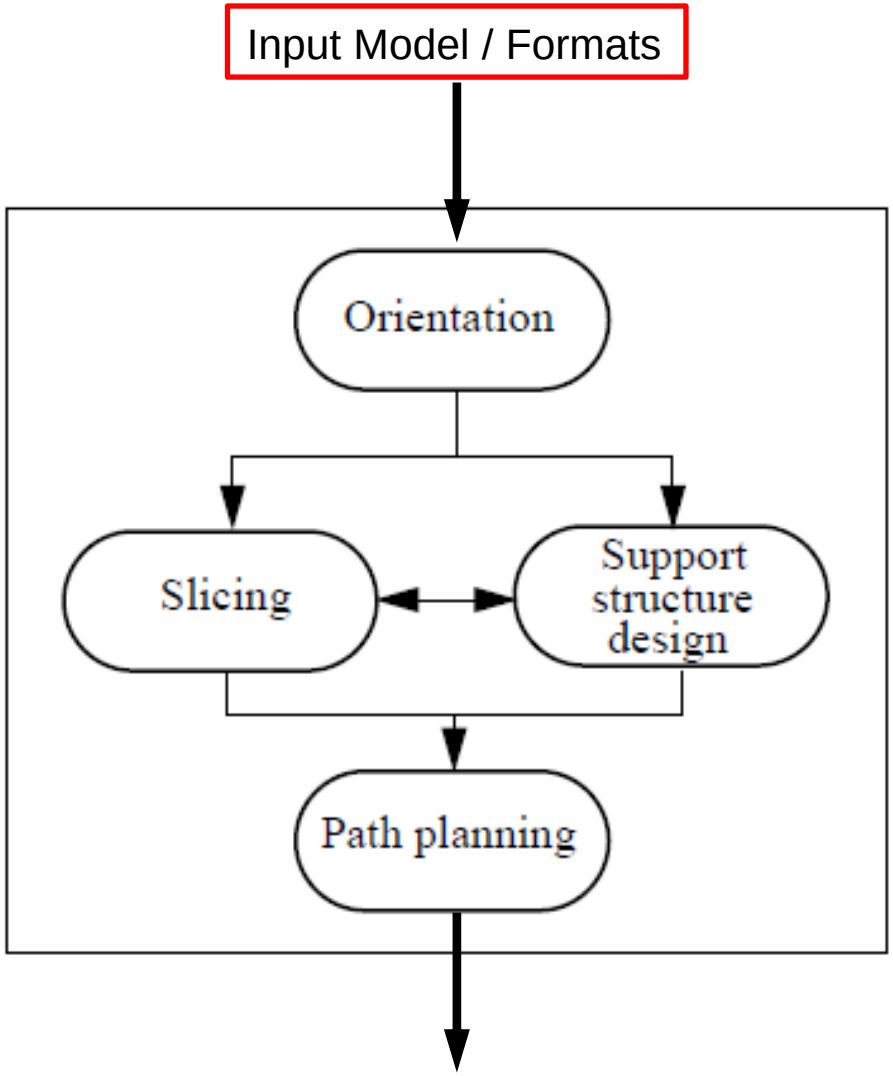
Stelian Coros

Plan for Today

- 3D Printing Process
  - Input Model
  - Orientation Determination
  - Support Structure Determination
  - Slicing
  - Path Planning
  - Machine Instructions

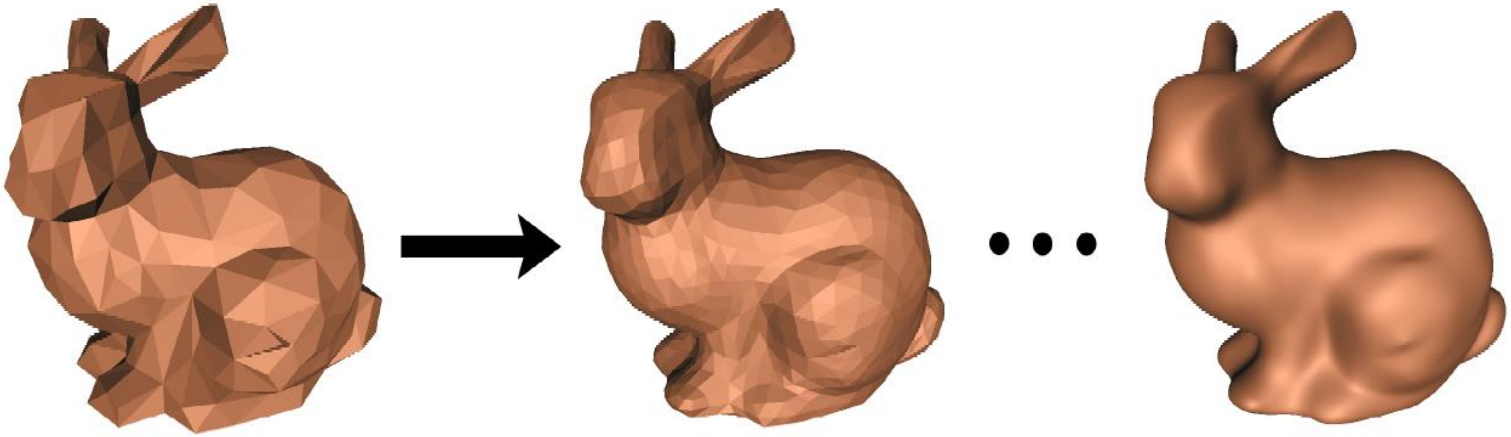


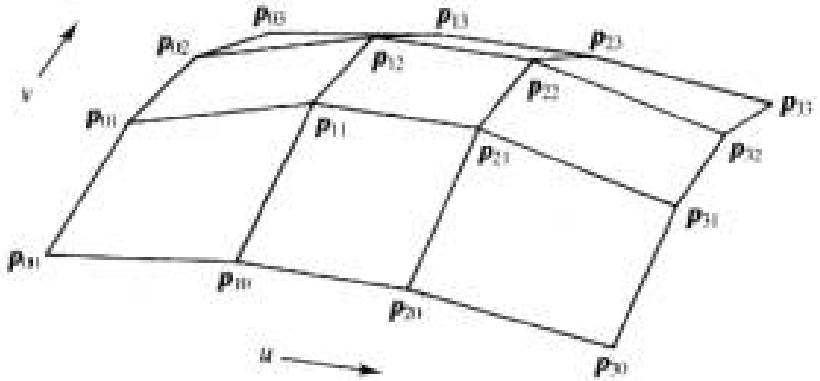
3D Printing Process

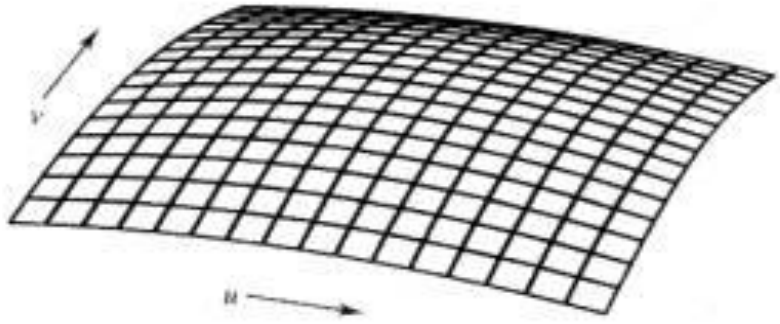


Specifying Input Models

- Surface models (i.e. triangle mesh, subdivision surfaces, etc)

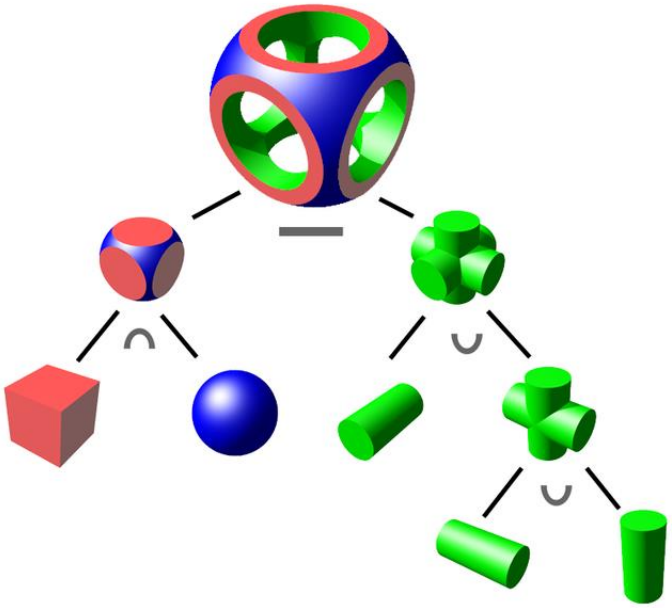


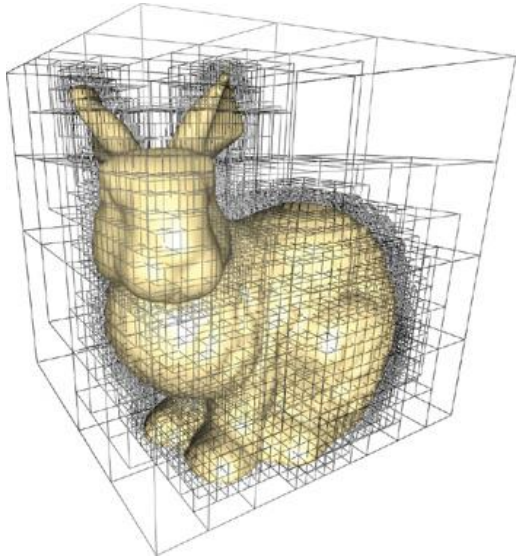


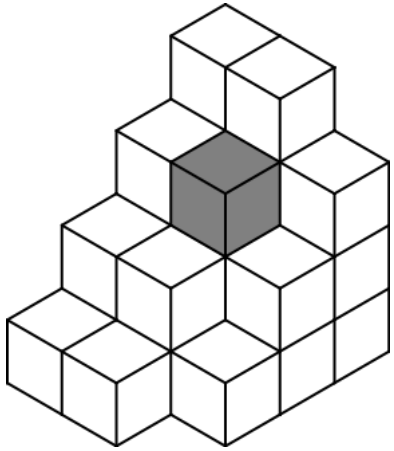


Specifying Input Models

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







STL (Stereolithography) File Format

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

ASCII

```
solid name

facet normal ni nj nk
    outer loop
        vertex v1x v1y v1z
        vertex v2x v2y v2z
        vertex v3x v3y v3z
    endloop
endfacet

endsolid name
```

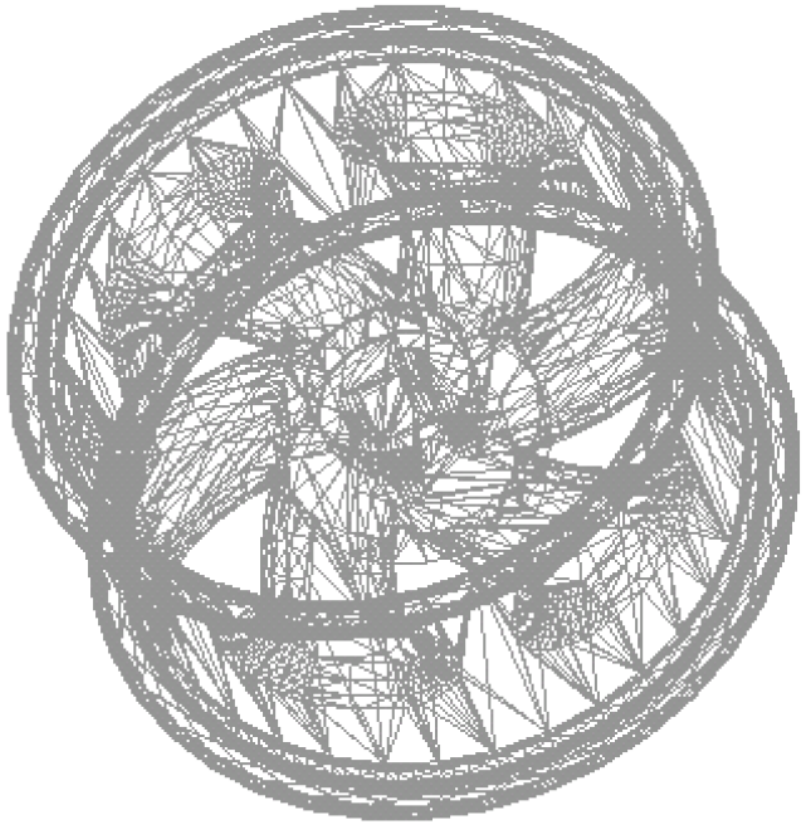
repeat for each triangle

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

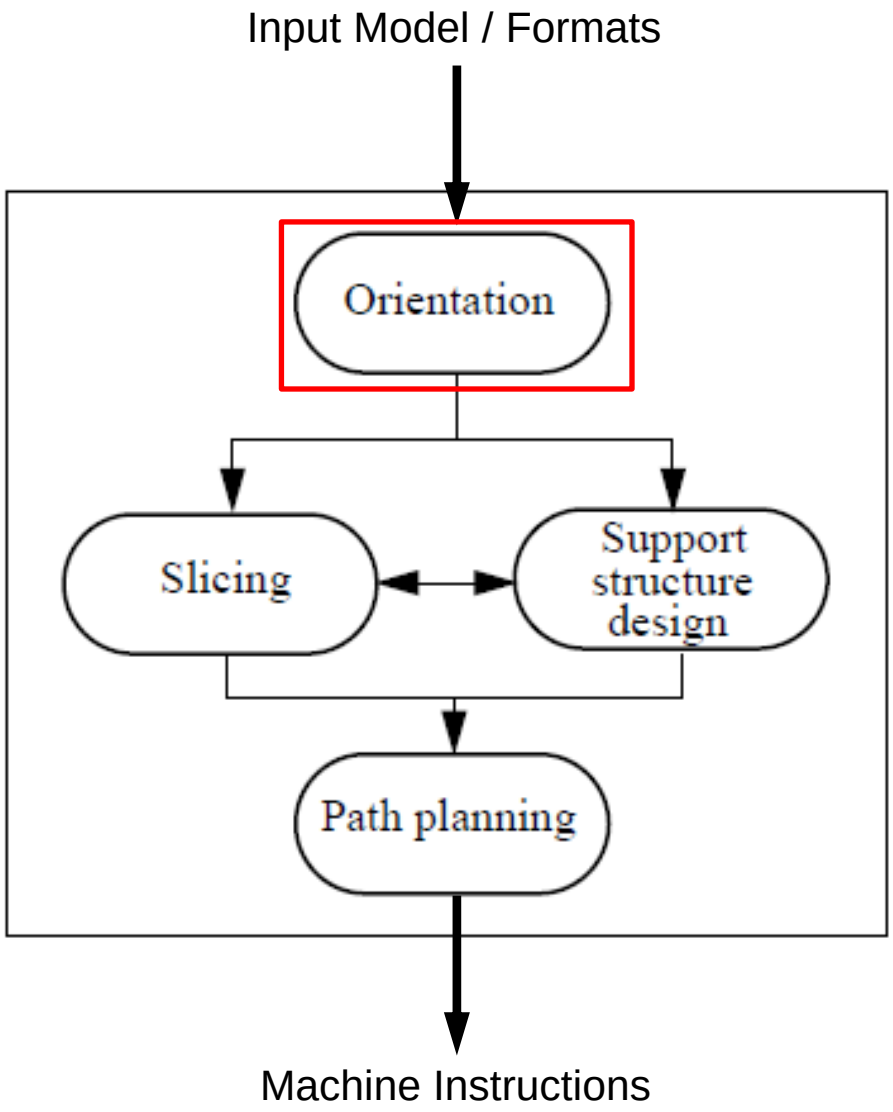
STL (Stereolithography) File Format



```
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
```

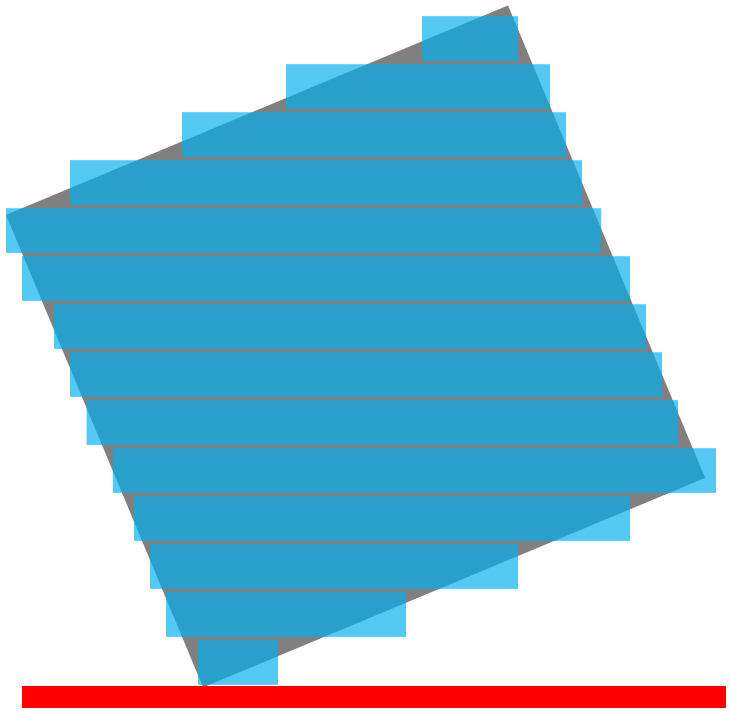
Source: Jackson 2000

3D Printing Process



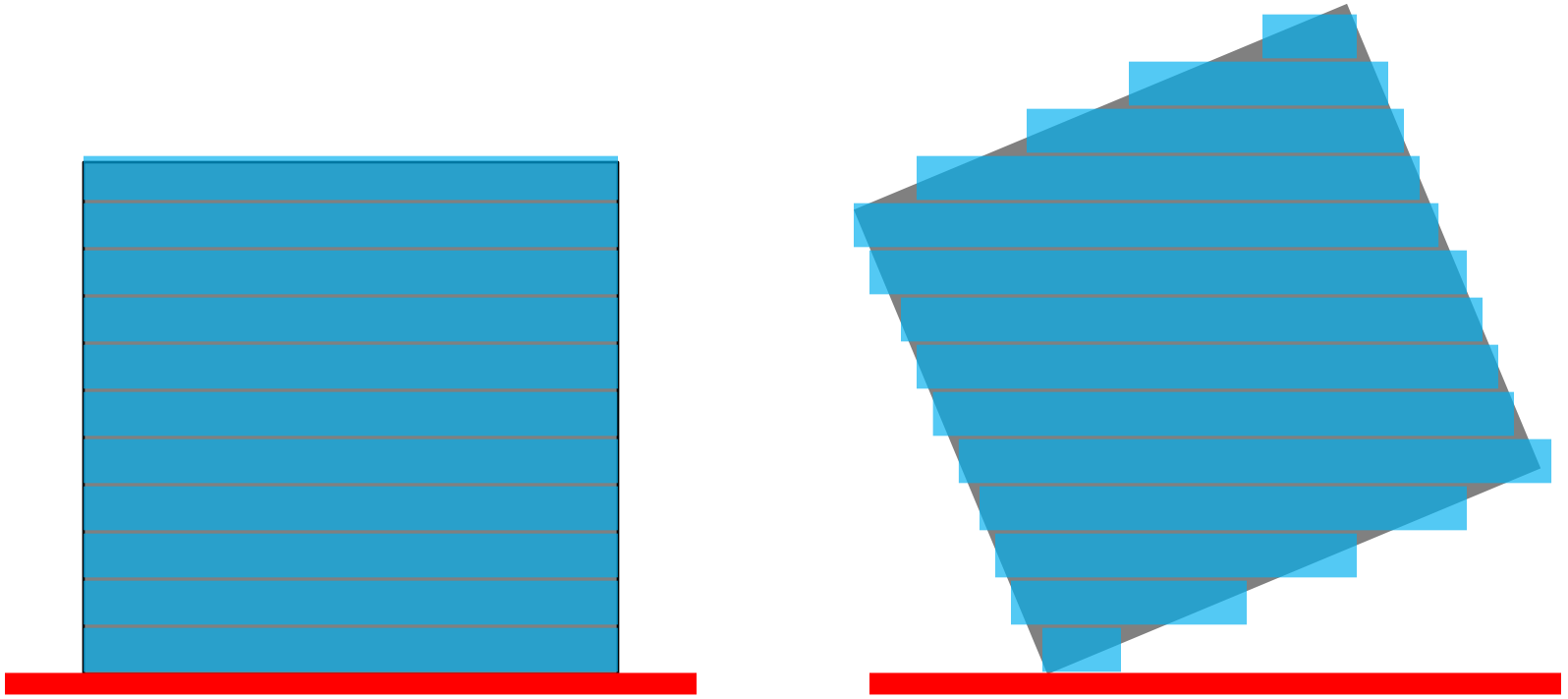
Orientation Determination

- Factors
  - Surface accuracy
  - Build time
  - Minimize support volume / contact area
  - Mechanical properties
  - Packing multiple objects
- Involves analysis of the 3D model



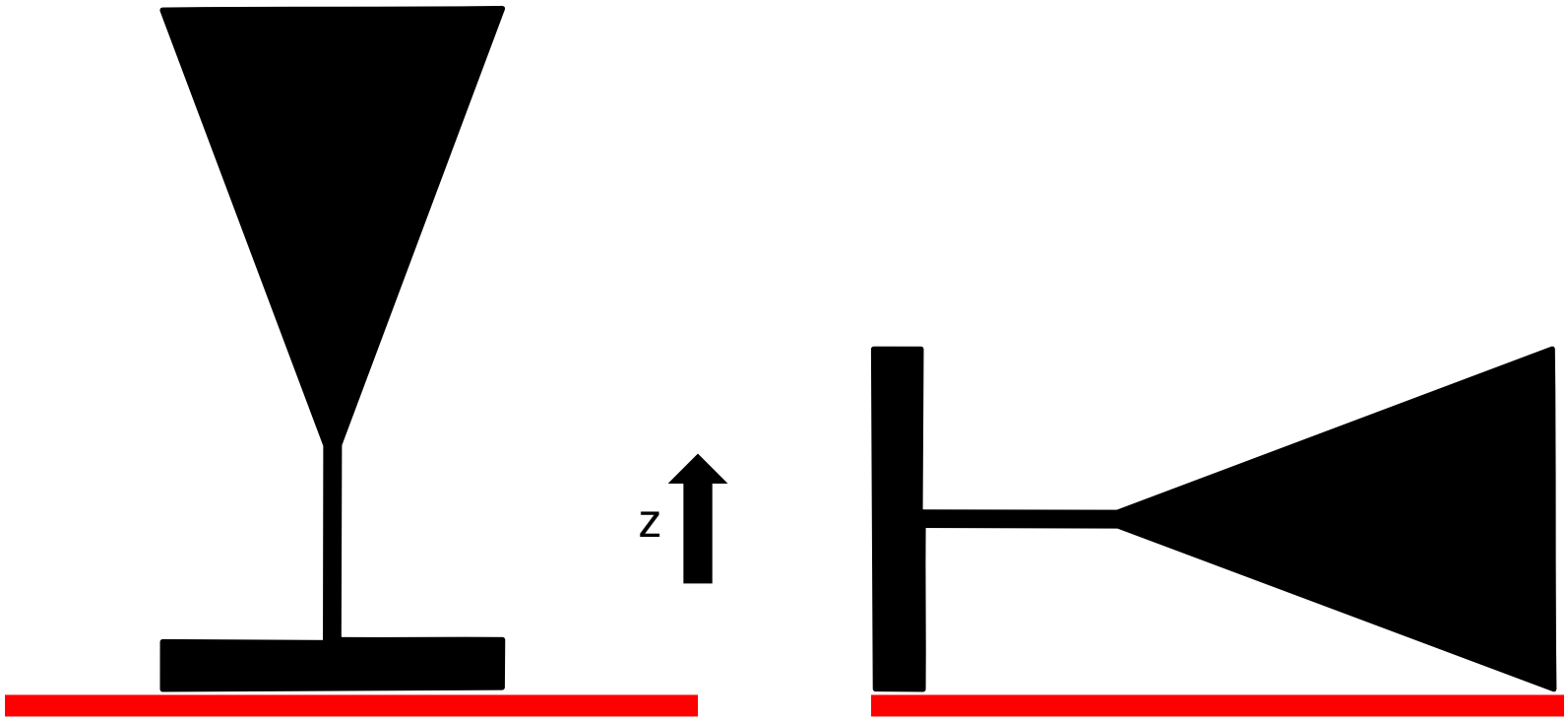
Orientation Determination: Surface Accuracy

- Minimize difference between input shape and printed result



Orientation Determination: Build Time

- Build speed is generally slower for the z direction compared to the xy direction

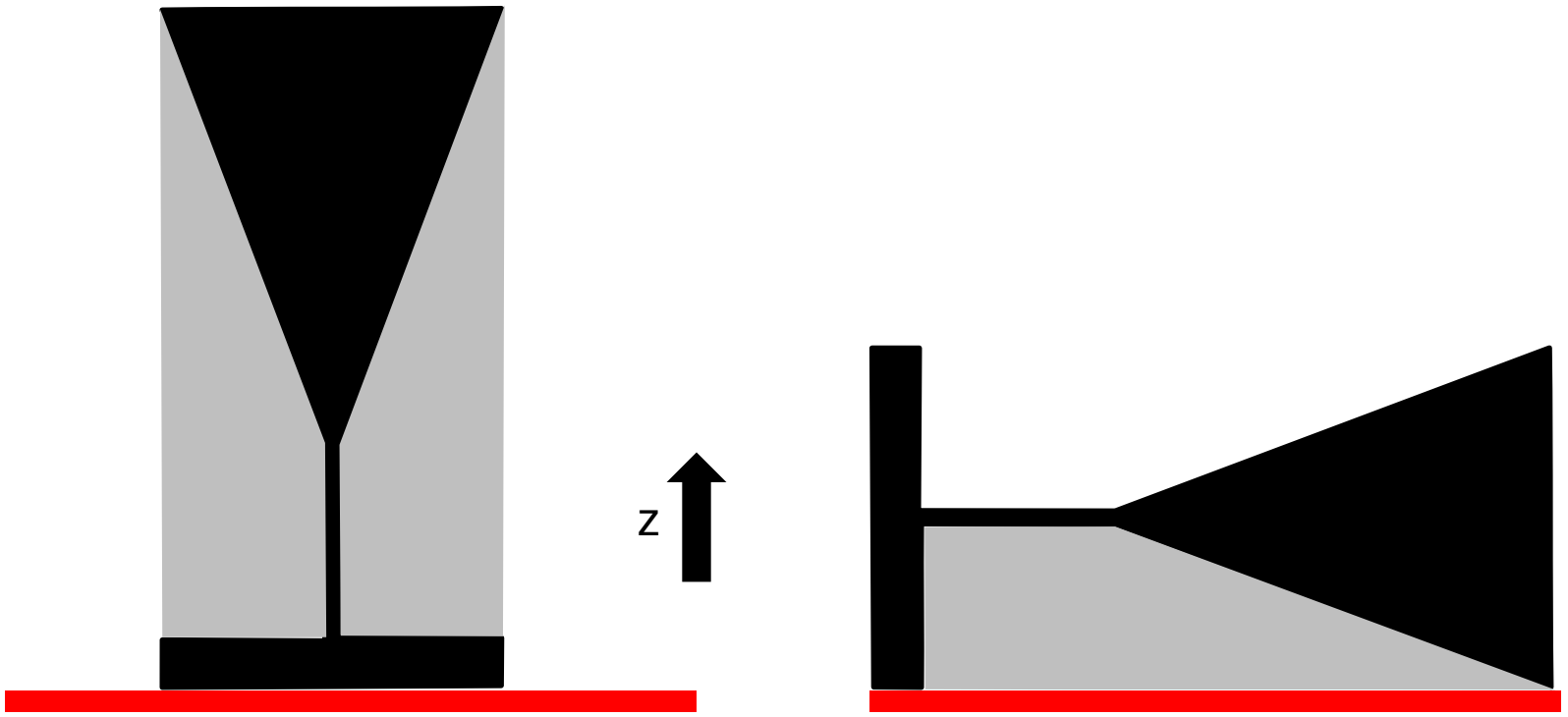


longer build time

shorter build time

Orientation Determination: Support Volume

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

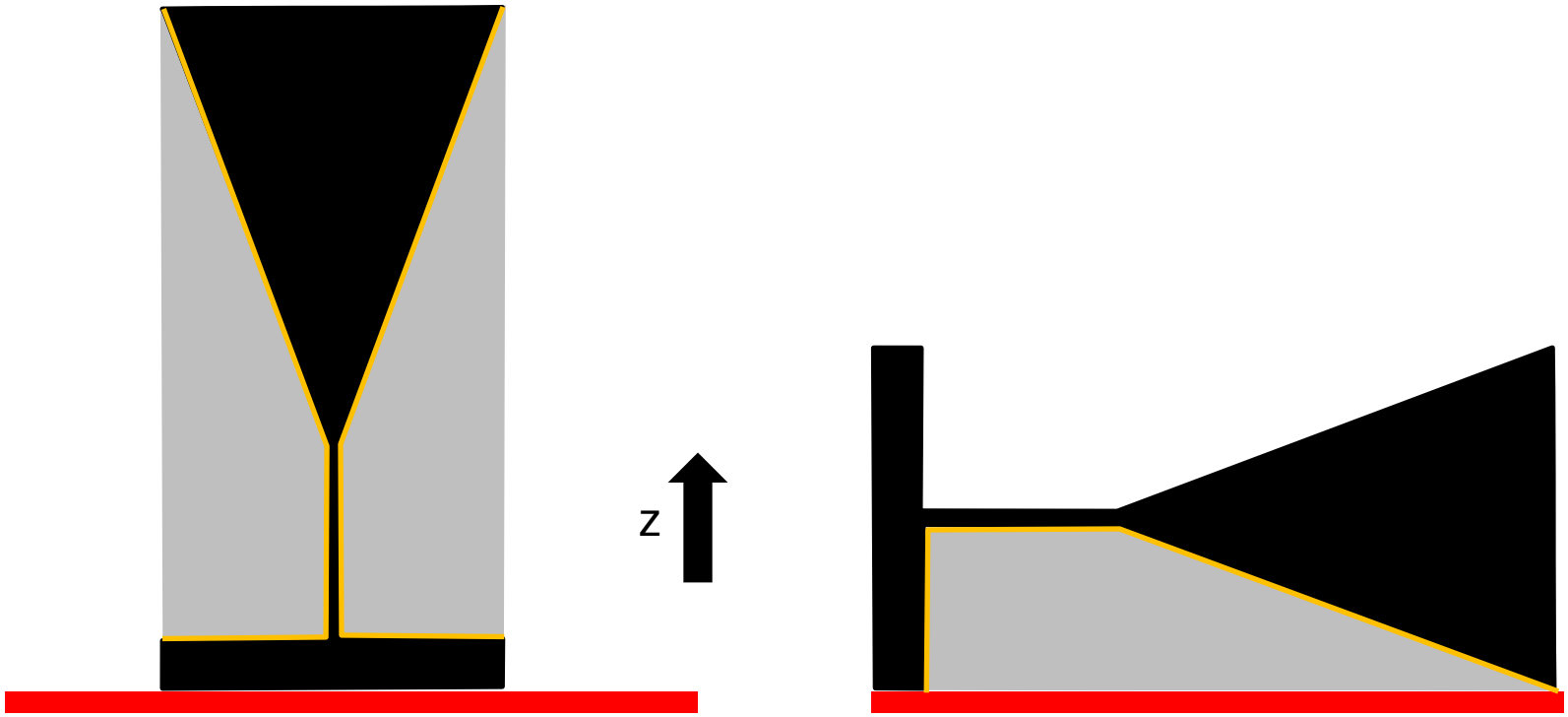


more support volume

less support volume

Orientation Determination: Support Contact Area

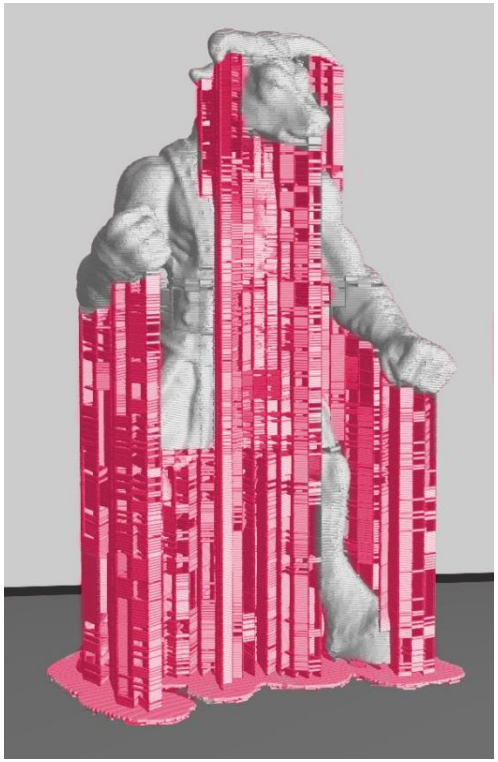
- Contact area between support material and object
  - Why?

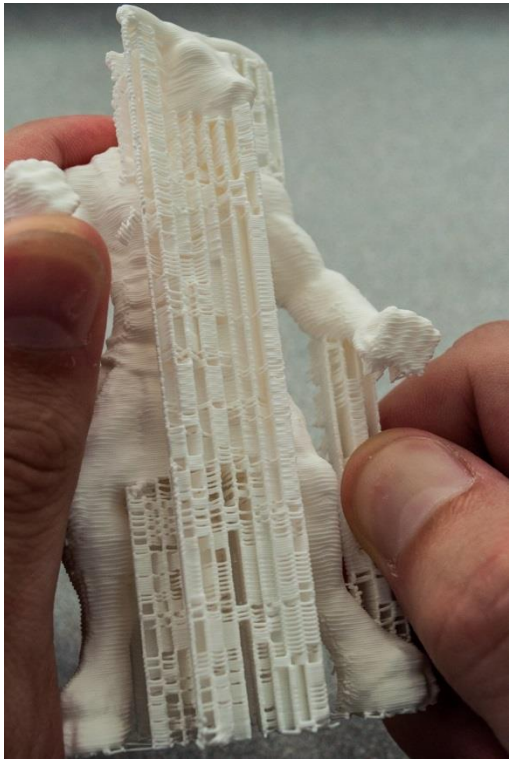


larger contact area

smaller contact area

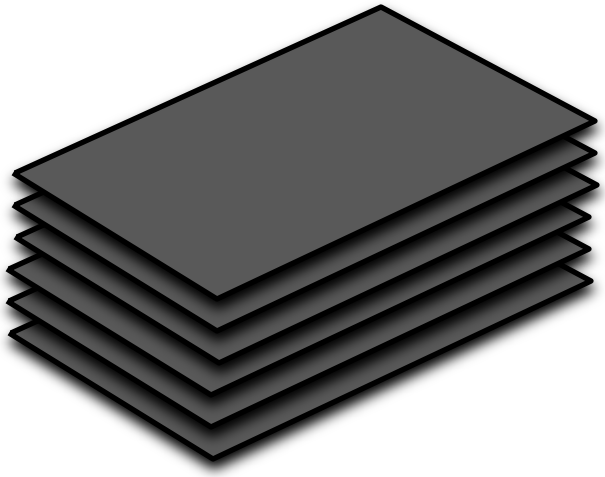
Orientation Determination: Support Contact Area





Orientation Determination: Mechanical Properties

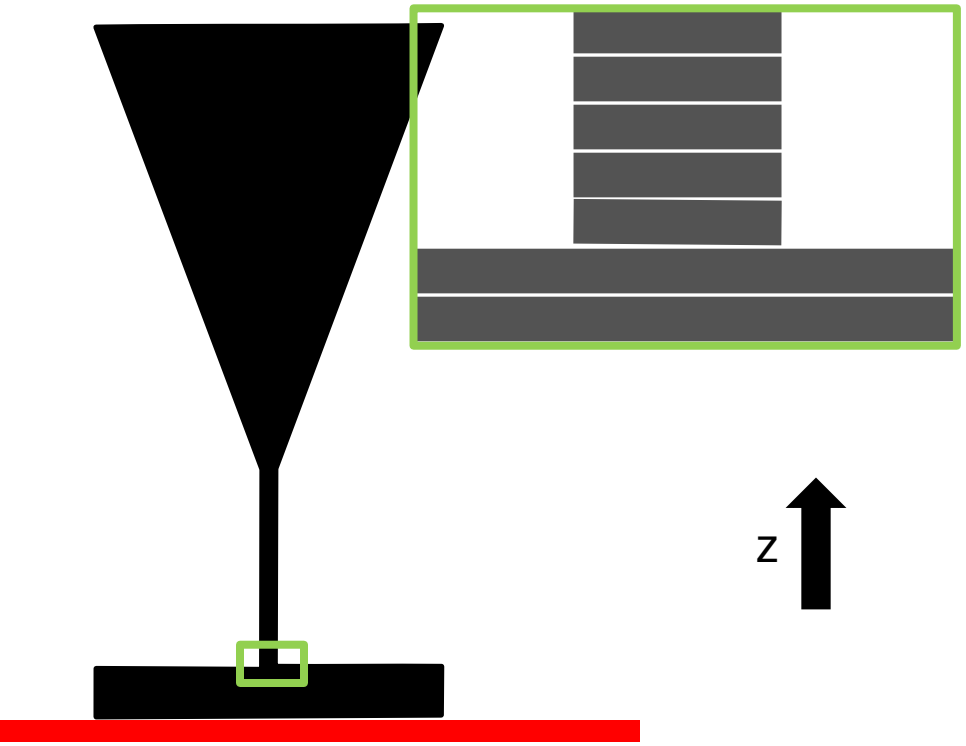
- Transversely isotropic materials
  - Mechanical properties (strength/stiffness) are isotropic within a layer but different across layers

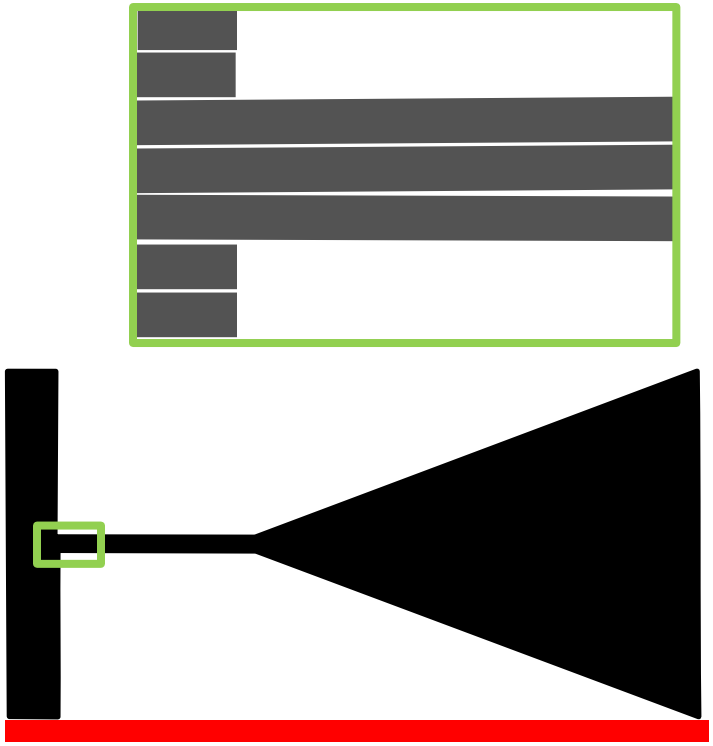


Source: http://en.wikipedia.org/wiki/

Orientation Determination: Mechanical Properties

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



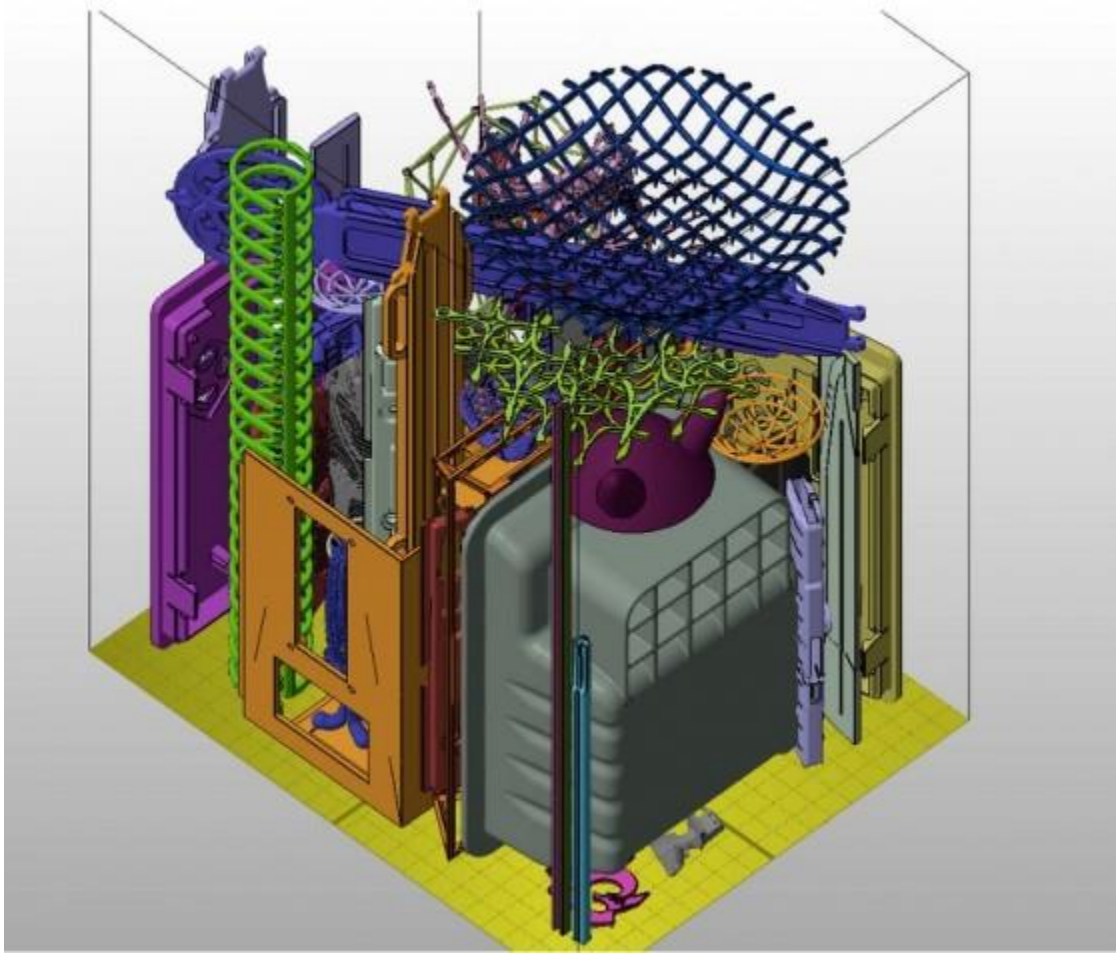


less strong features

stronger features

Orientation Determination: Packing

- Pack as many objects as possible within a print volume

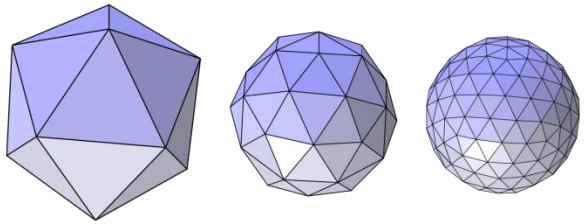


Typical Approaches for Orientation Determination

- 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
  - placement is computed using optimization according to one or more objectives (build time, support volume, support area, mechanical properties)

Algorithms to Compute Orientation

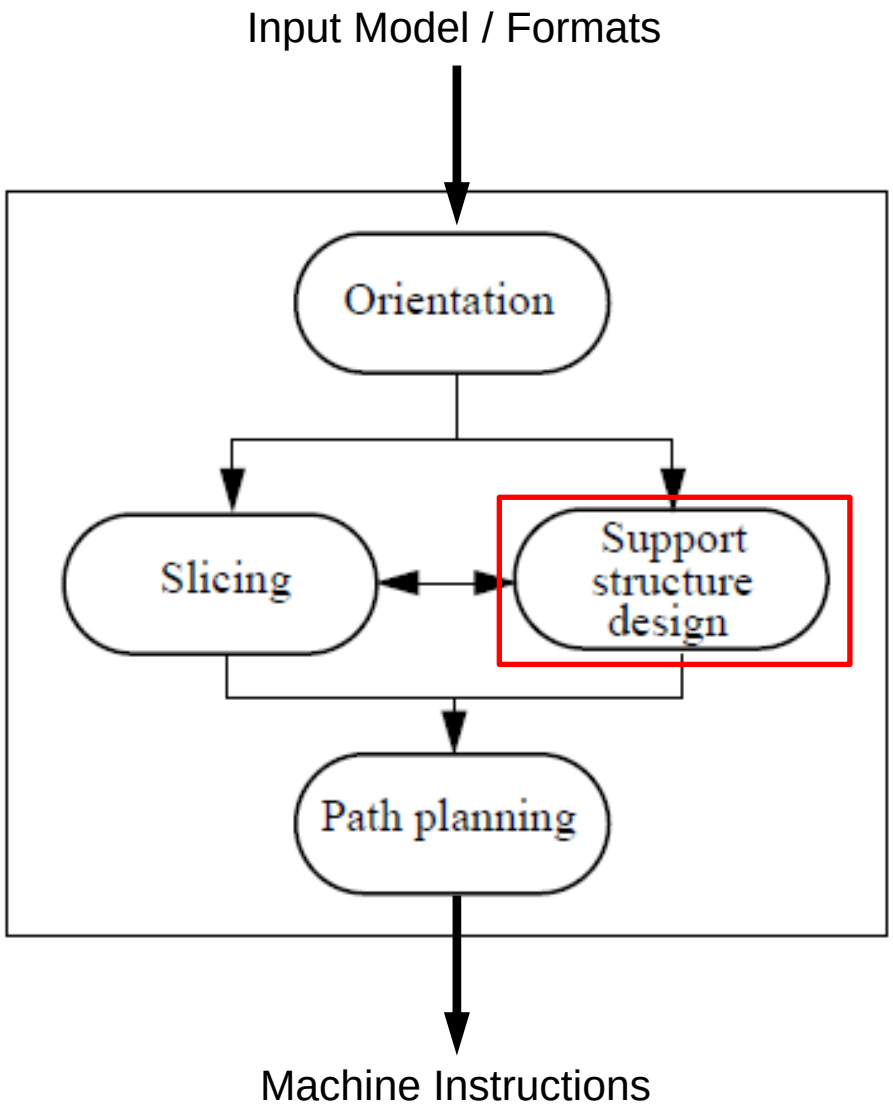
- Exhaustive search
- Compute a uniform set of directions
  - Icosahedron subdivision



- Optimization-based
  - Compute objective as a function of orientation
    - 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)

Possible project topic!

3D Printing Process

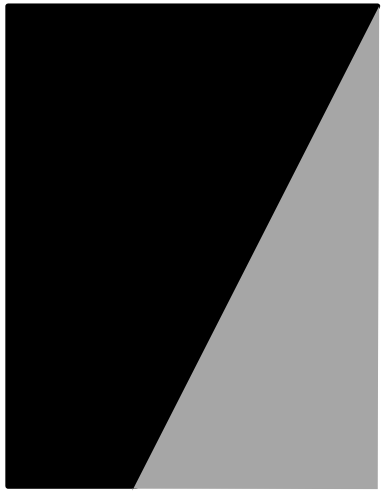


Support Structure Design

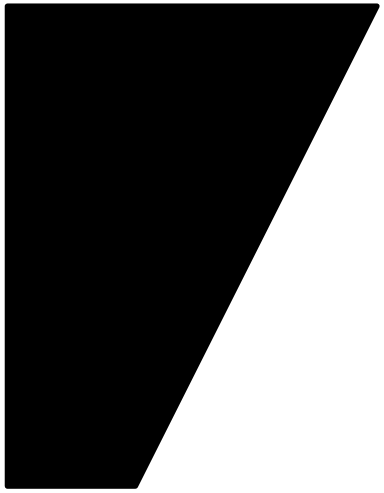
- Do not require support
  - SLS, DMLS, LOM, Plaster-based
- Require support
  - SLA, FDM, phase-change inkjet
- Different goals
  - Supporting overhangs
  - Prevent curling as materials harden
  - Maintaining stability (part does not move, tip over)
  - Supporting large flat walls

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 Structure Design

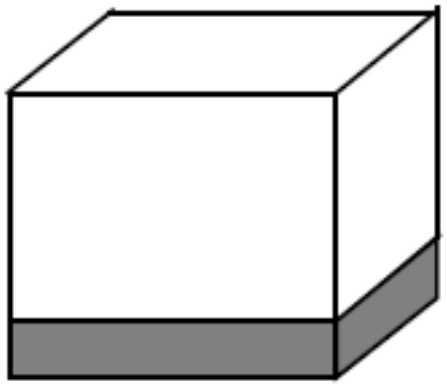
- Step 1: compute support volume
- Step 2: fill support volume with appropriate structure

- Want: easy to remove structures
- Do not want: significantly increase material usage or printing time

Support Types

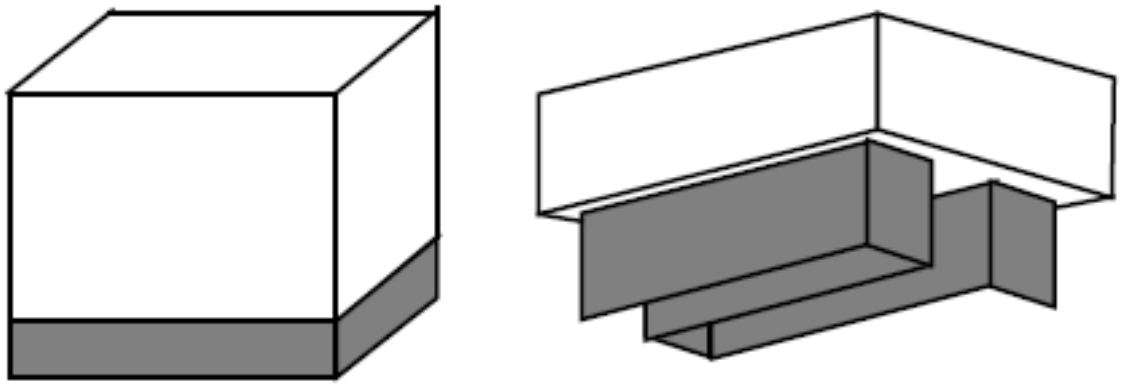
Support Types

- Base support



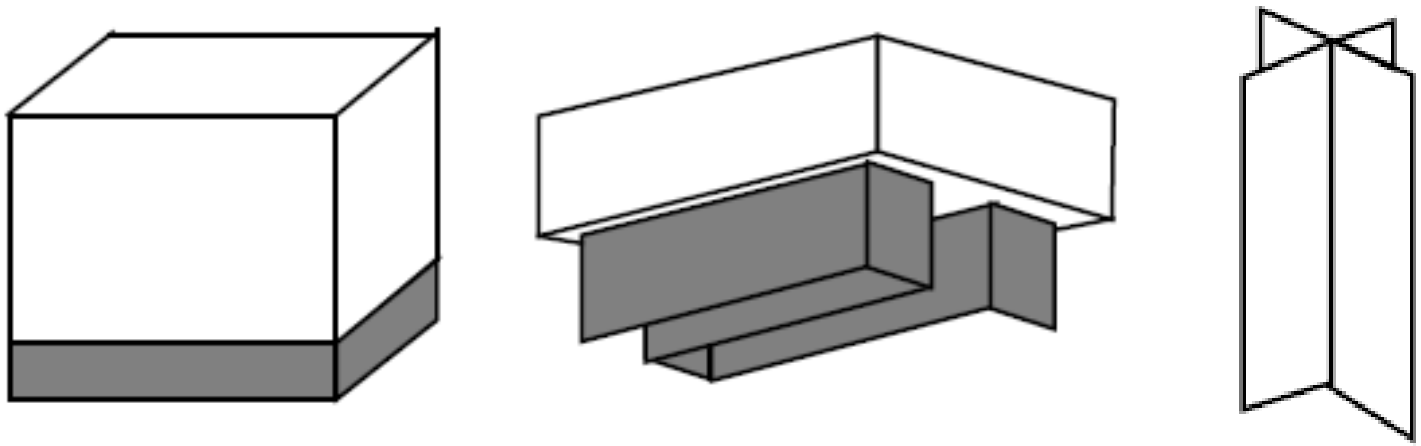
Support Types

- Base support
- Zigzag support (FDM)



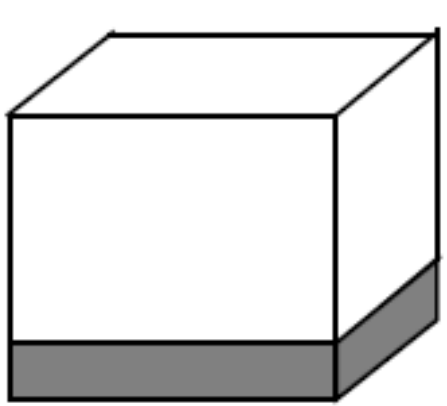
Support Types

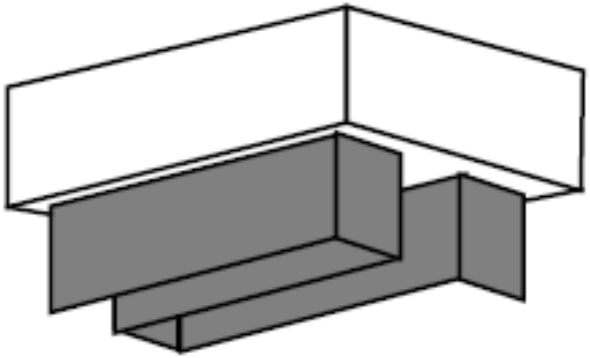
- Base support
- Zigzag support (FDM)
- Column support (SLA)

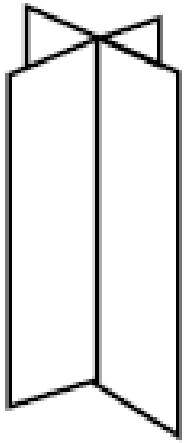


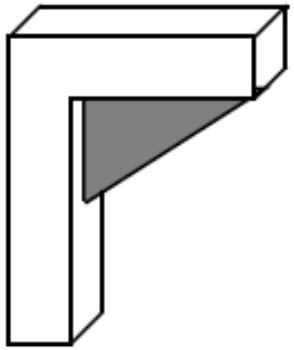
Support Types

- Base support
- Zigzag support (FDM)
- Column support (SLA)
- Gusset support



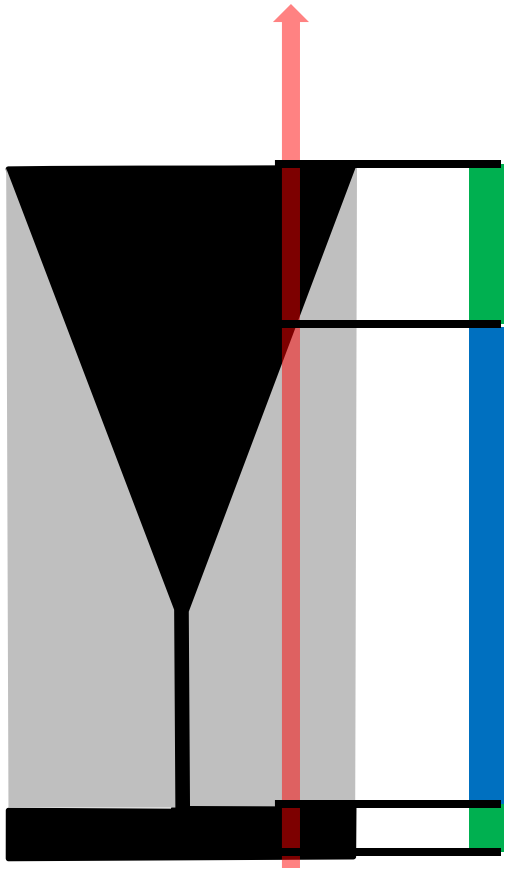






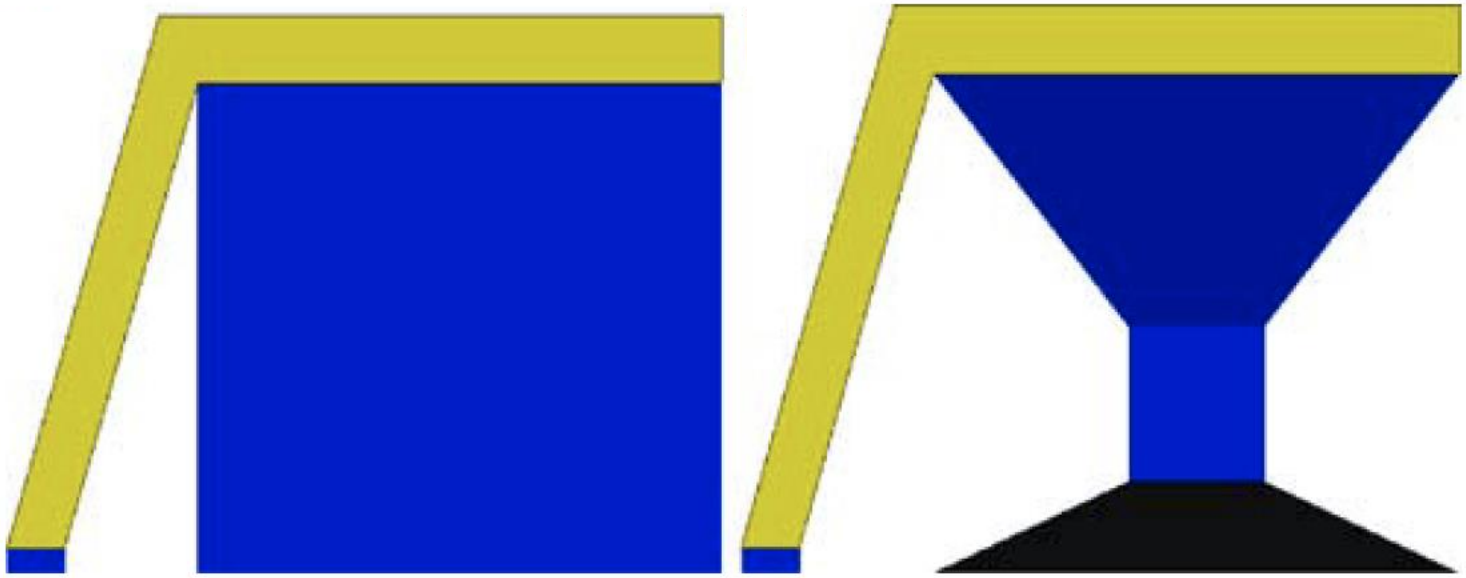
Support Volume: Simple Conservative Algorithm

- 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



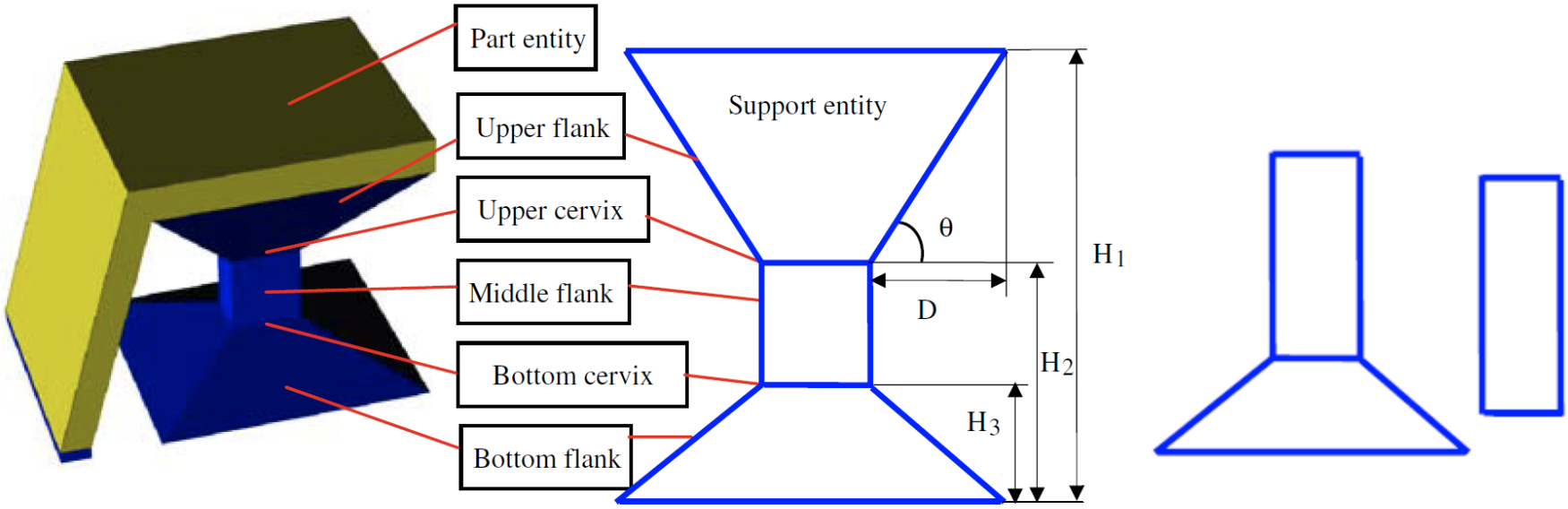
straight wall support

slant wall support

Huang et al. 2009

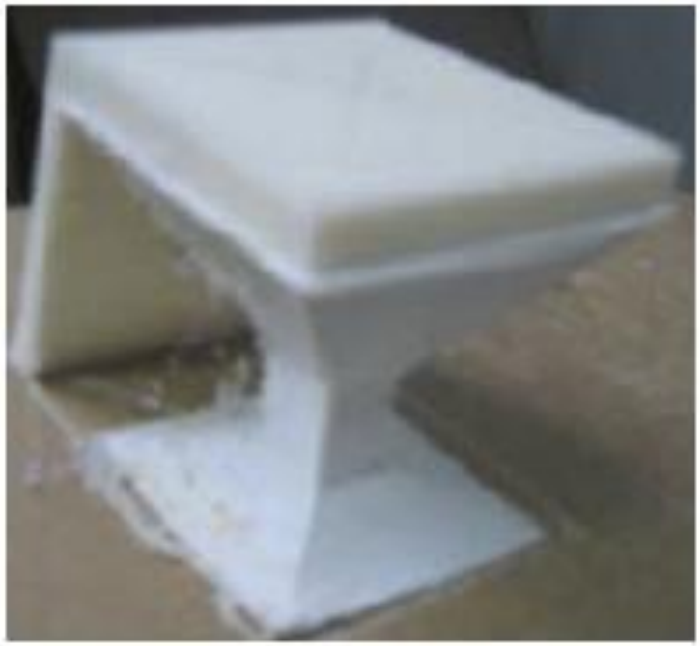
Advanced Algorithms

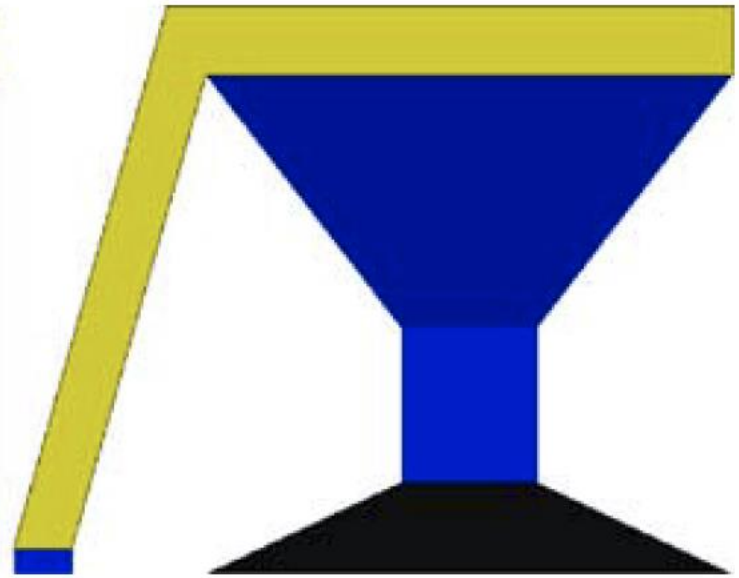
- Minimize the use of support material



Huang et al. 2009

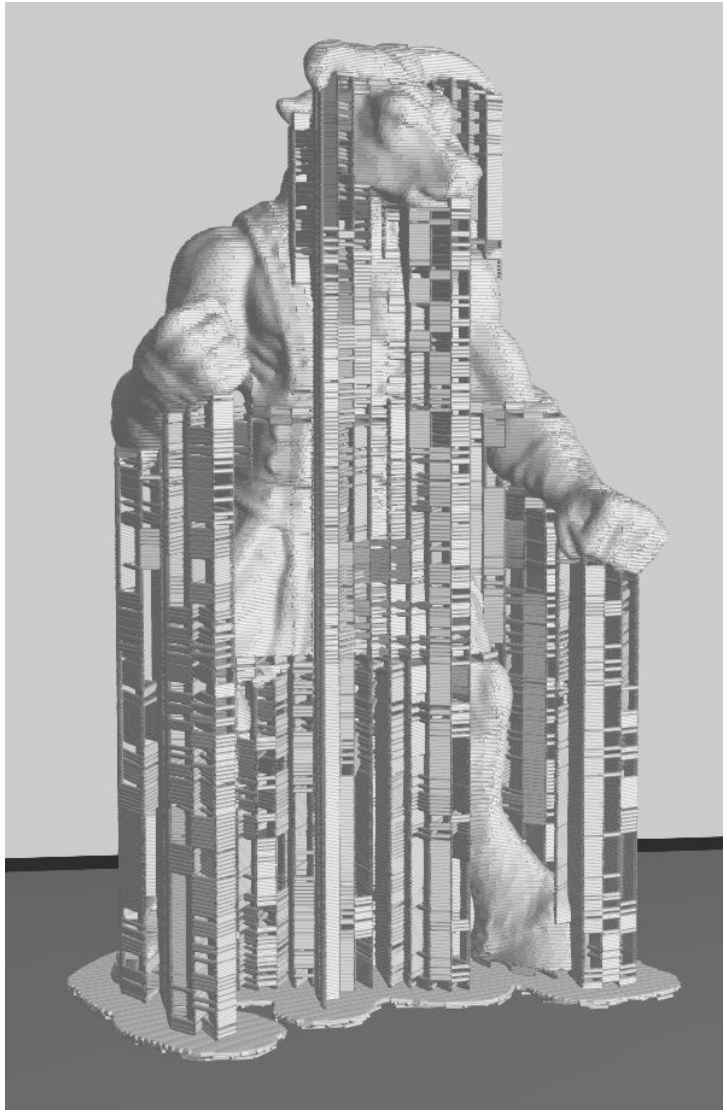
Manufactured Result





Huang et al. 2009

Advanced Algorithms: Makerware



Advanced Algorithms: Meshmixer

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

Support Structure Generation: Ongoing Research

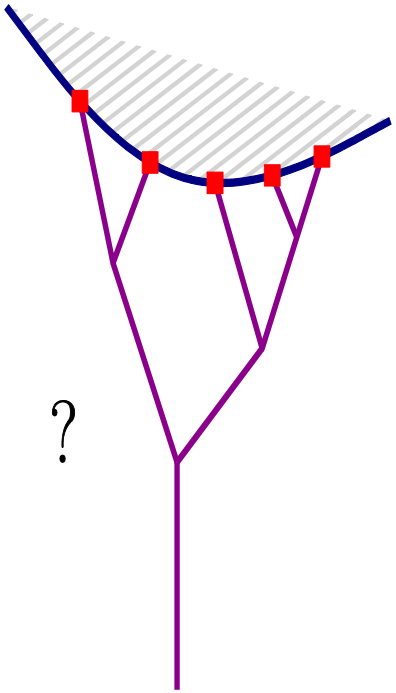
Bridging the Gap: Automated Steady Scaffoldings for 3D Printing

Jérémie Dumas, Jean Hergel and Sylvain Lefebvre

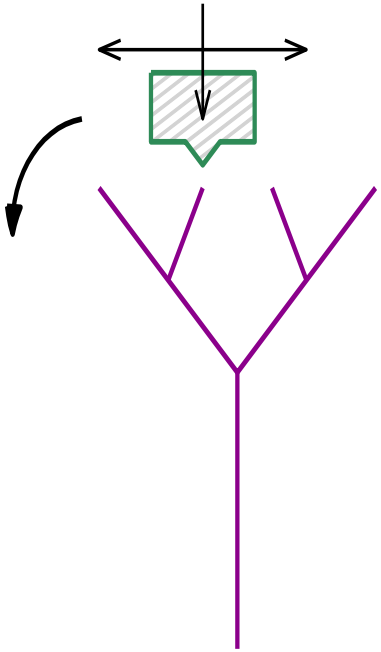
SIGGRAPH 2014

Trees - Pros and Cons

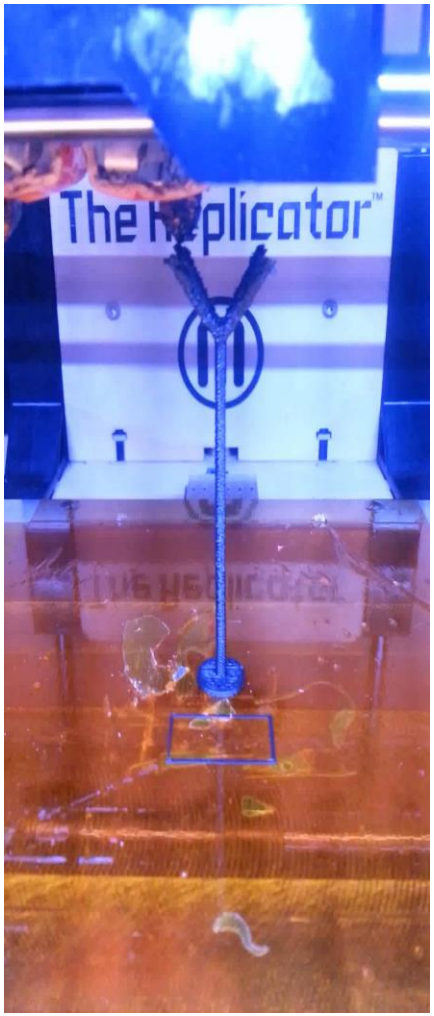
Good Support/length ratio



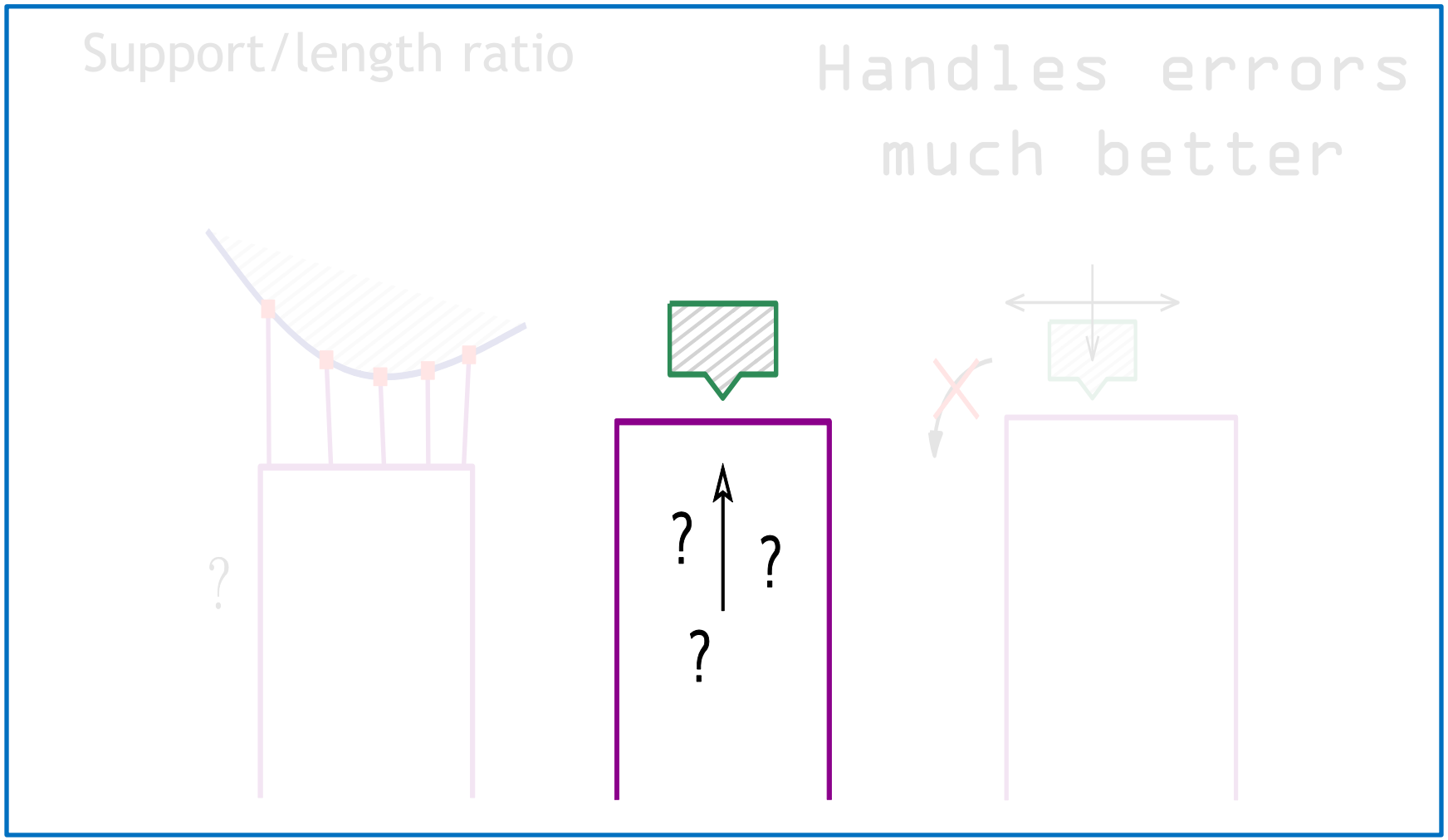
Very sensitive to small errors - unstable



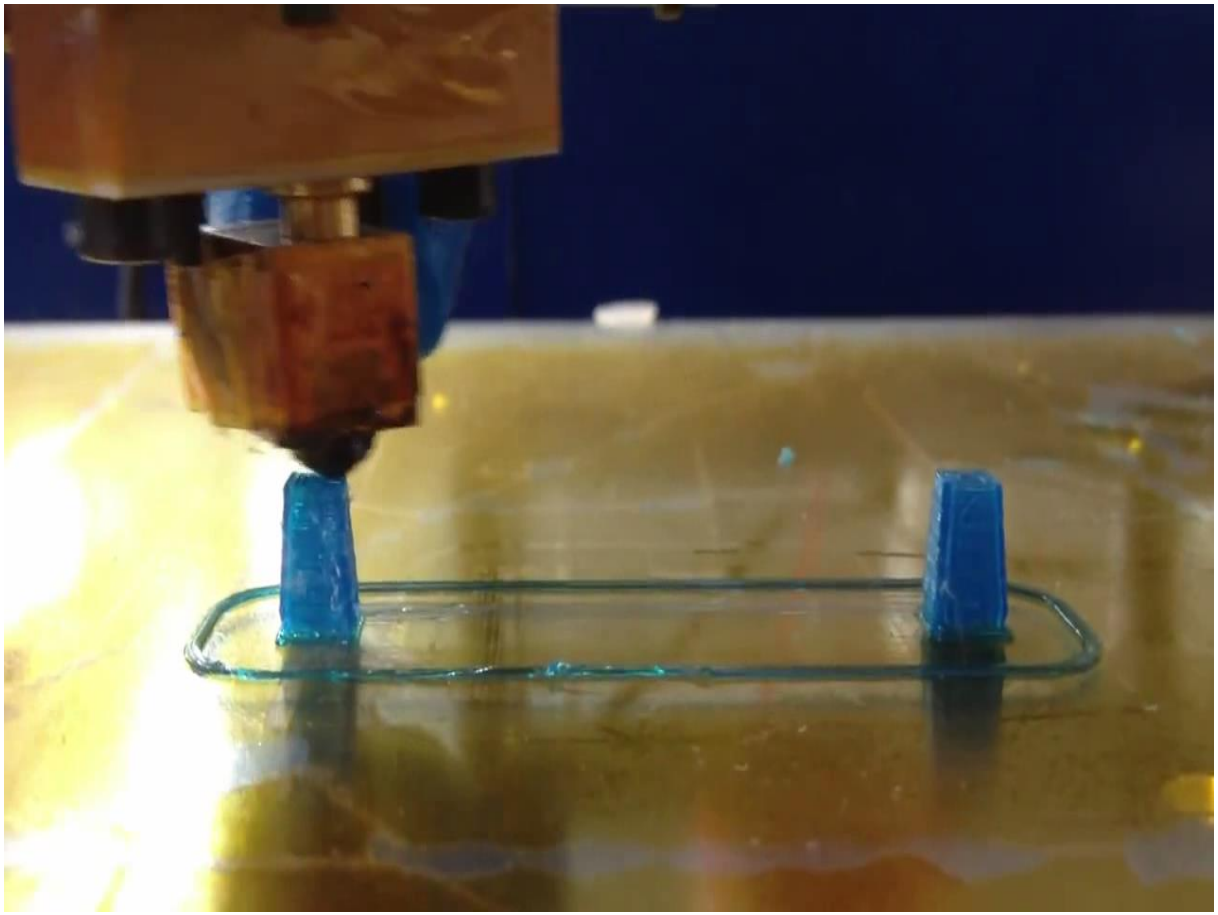
Unstable structures



Think Bridges

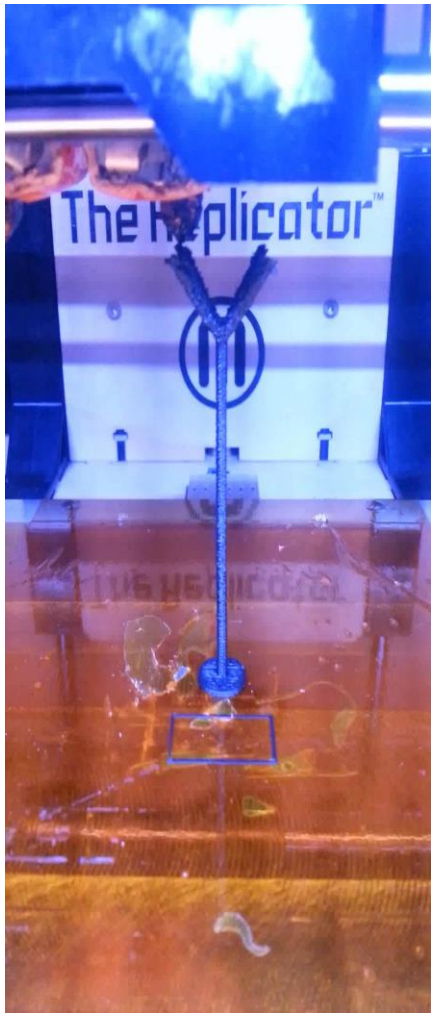


Printing Bridges



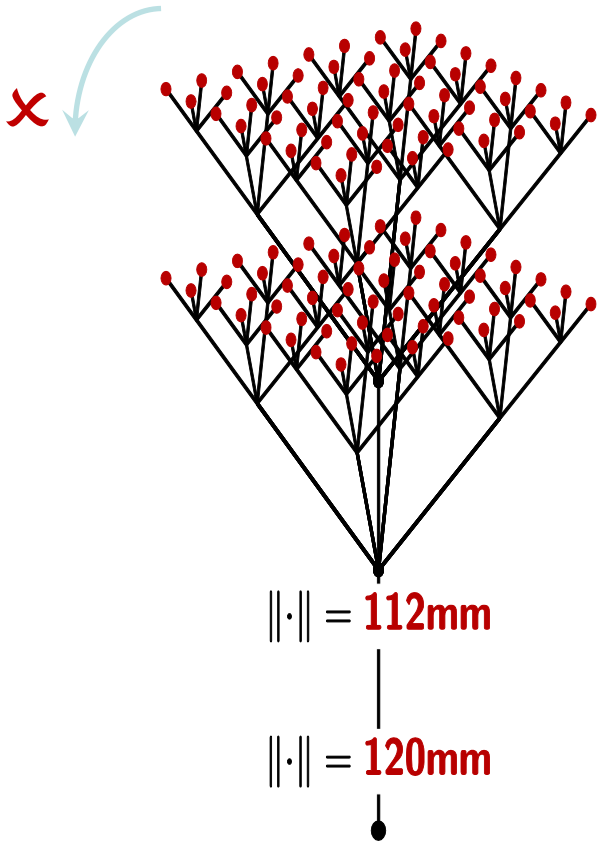
Source: http://youtu.be/wK2APNwEoSk

Torque

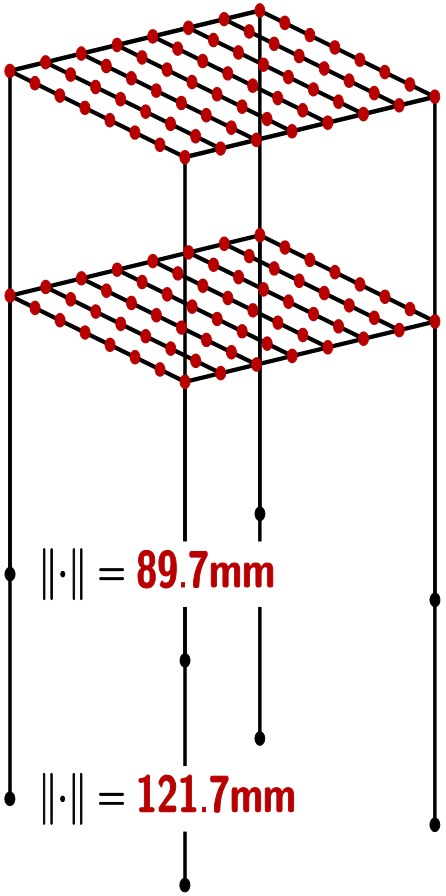


Material usage

Competitive to trees

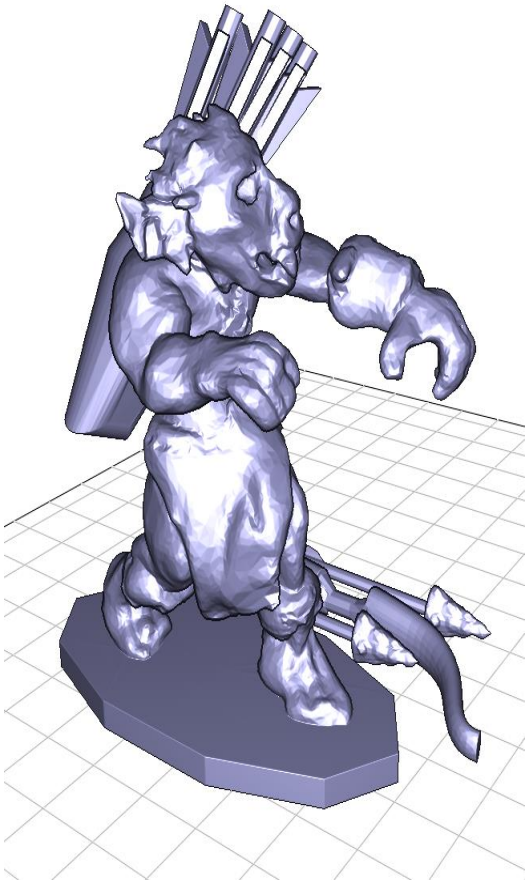


…Up to a certain height



Method Overview

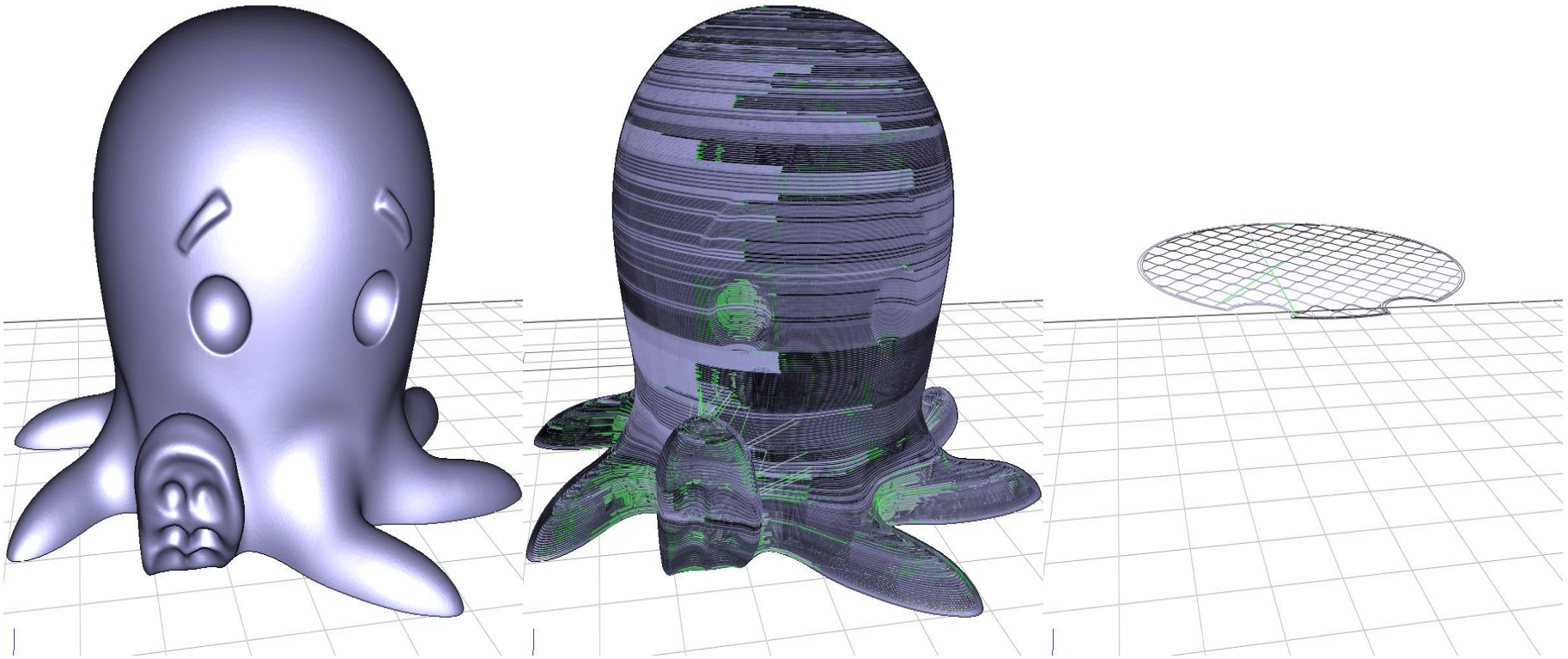
1. Overhang detection
2. Bridge (scaffolding) synthesis



thing:347046

Overhang Detection

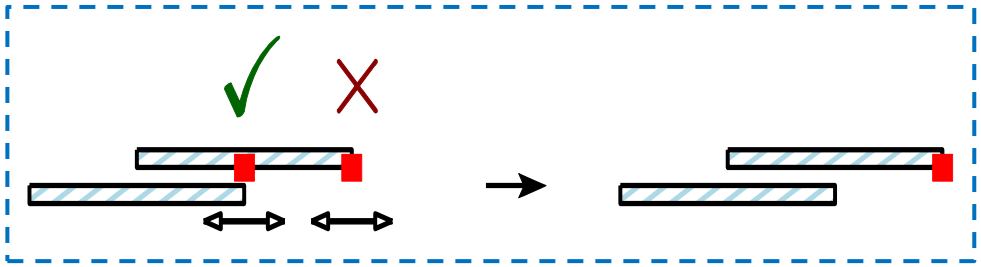
- Input: 3D Model + Slices + Print head paths

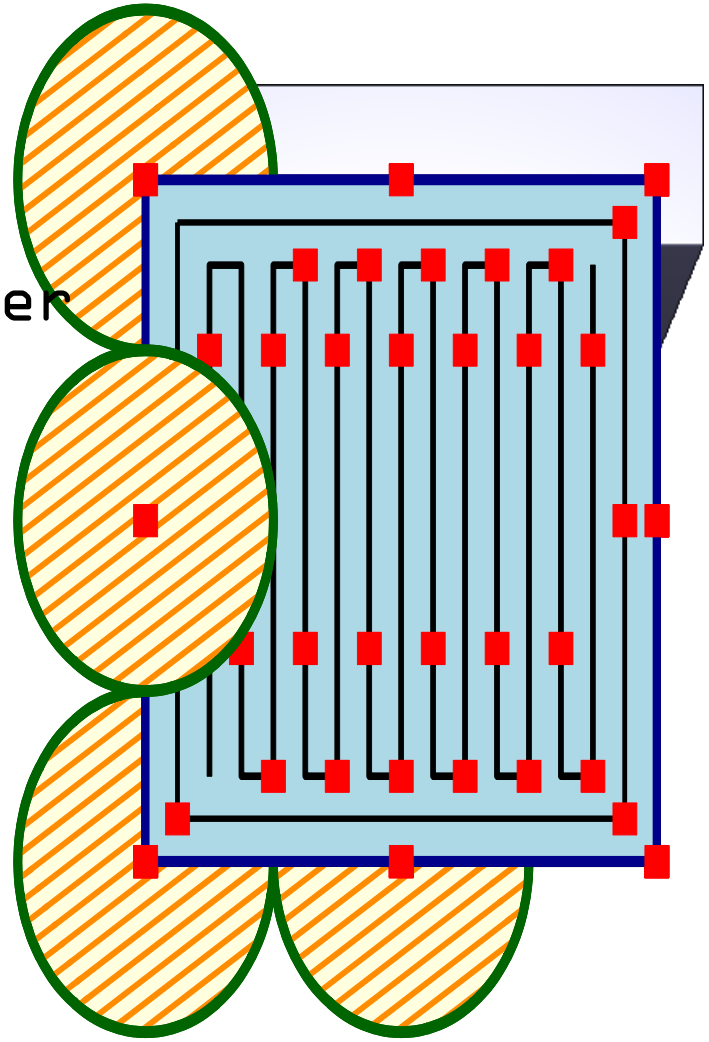


thing:27053

Overhang Detection

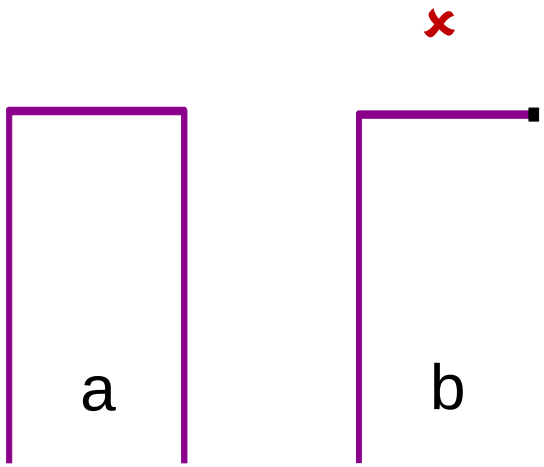
1. Use print paths
2. Uniform sampling
   - Need Support ? Y / N
3. Simplify wrt print order



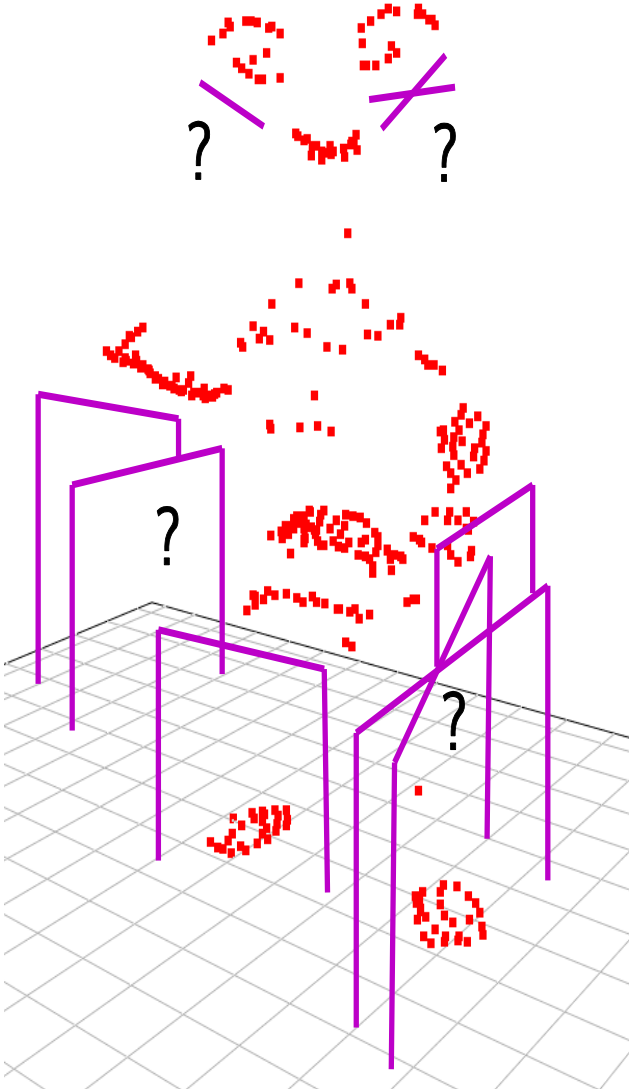


Scaffolding: Problem Statement

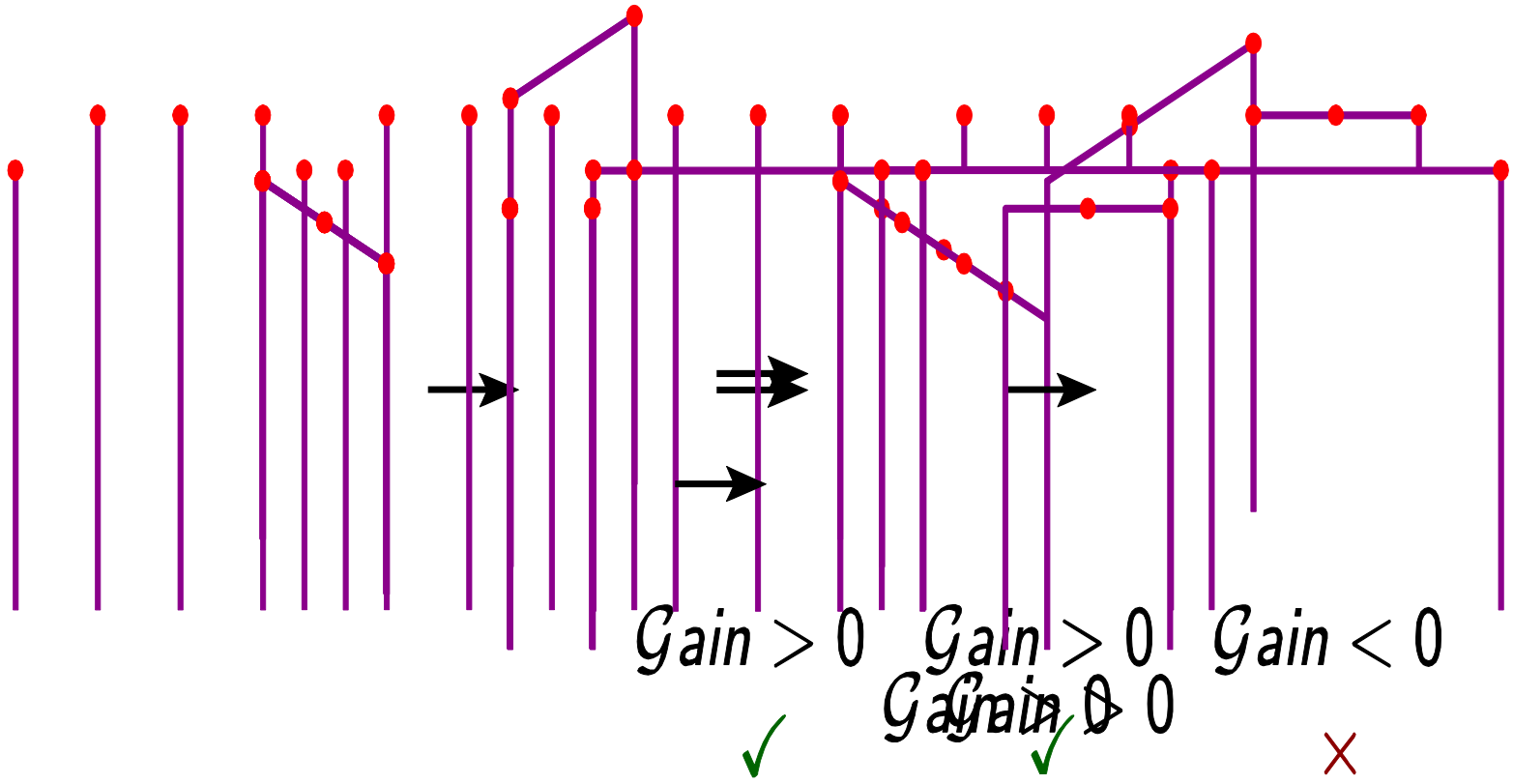
- In - Required Points
- Out - Valid Scaffolding



→ Global constraint

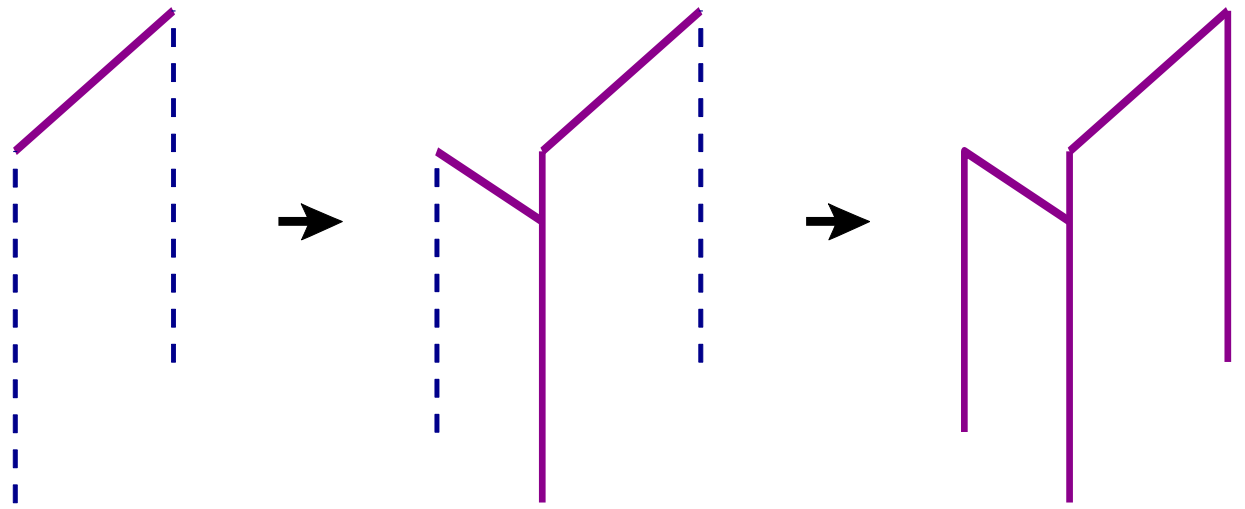


Goal: *Minimal Length* Scaffolding



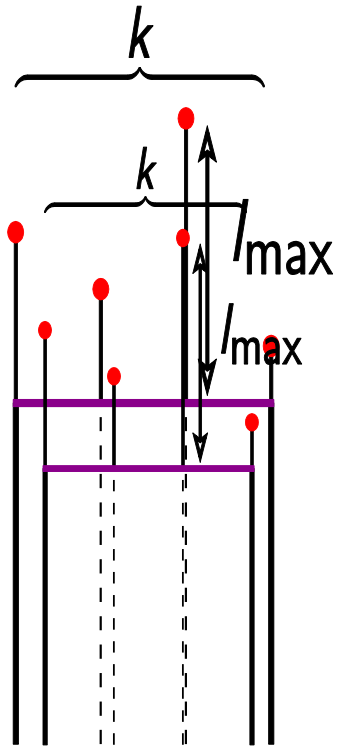
Our Approach

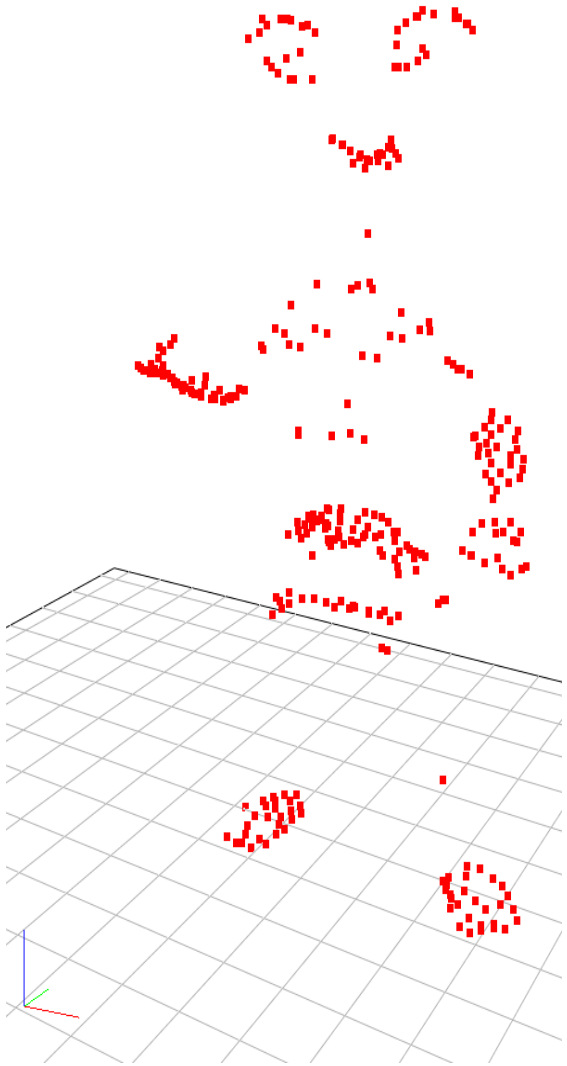
- Finding a global optimum is not easy
- Heuristic Greedy Algorithm



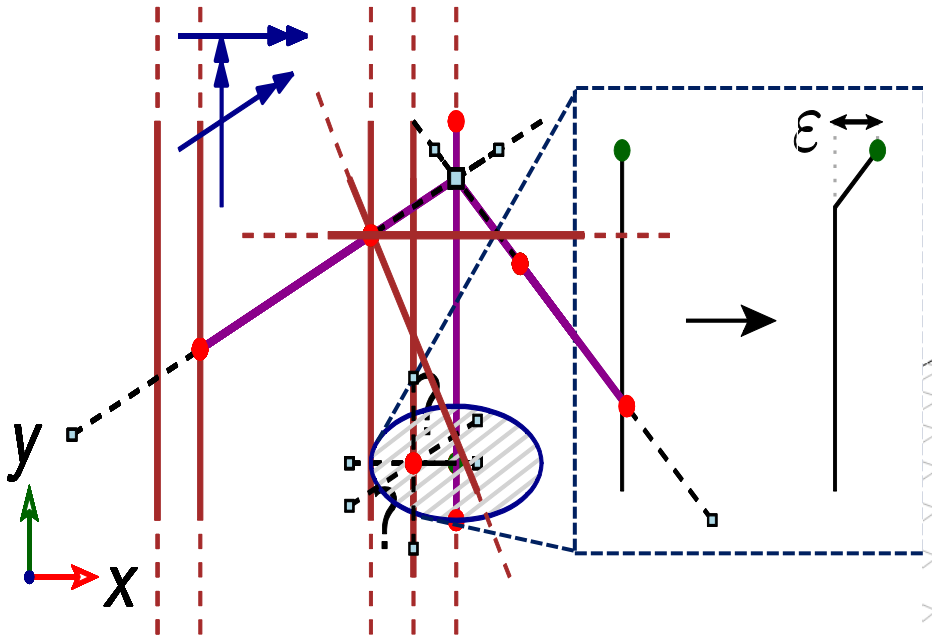
Algorithm Overview

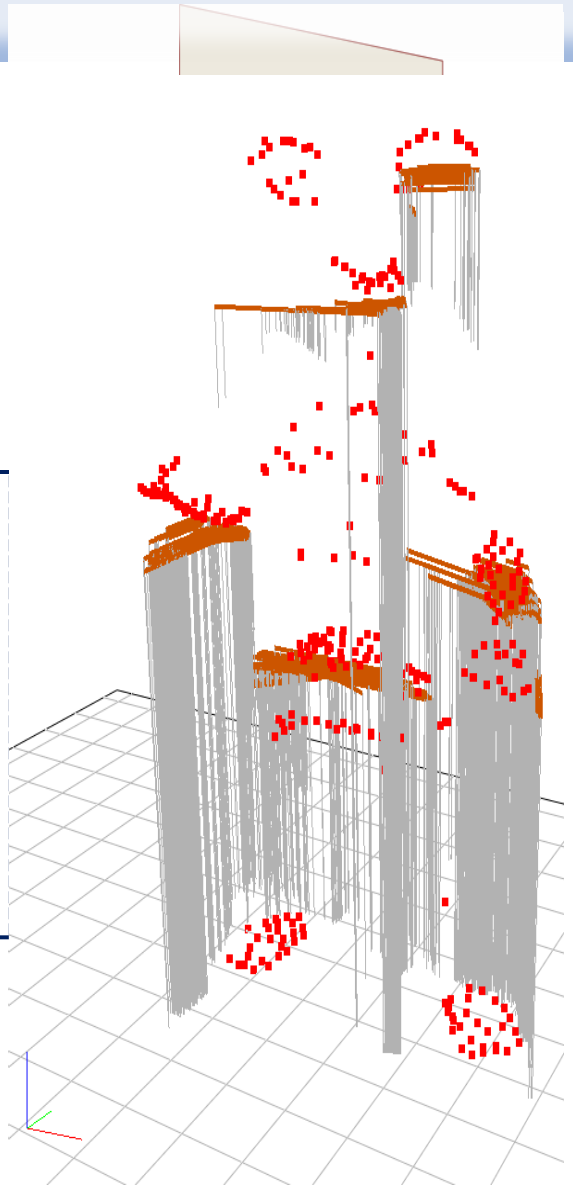
1. Required Points
2. Candidate Bridges
3. Greedy Selection
4. Repeat
5. Stop



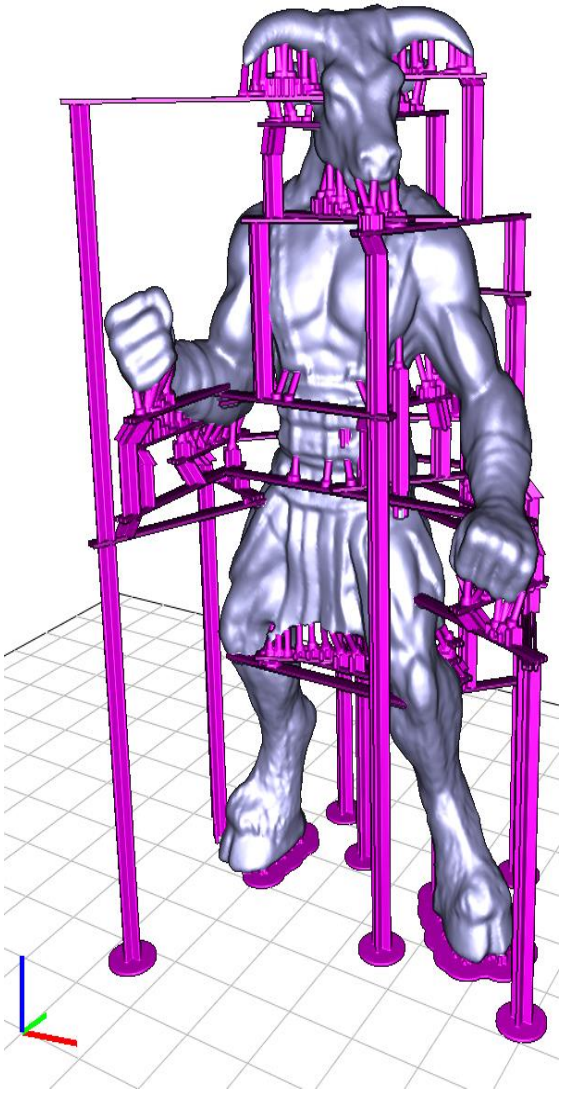


Candidate Bridge Generation

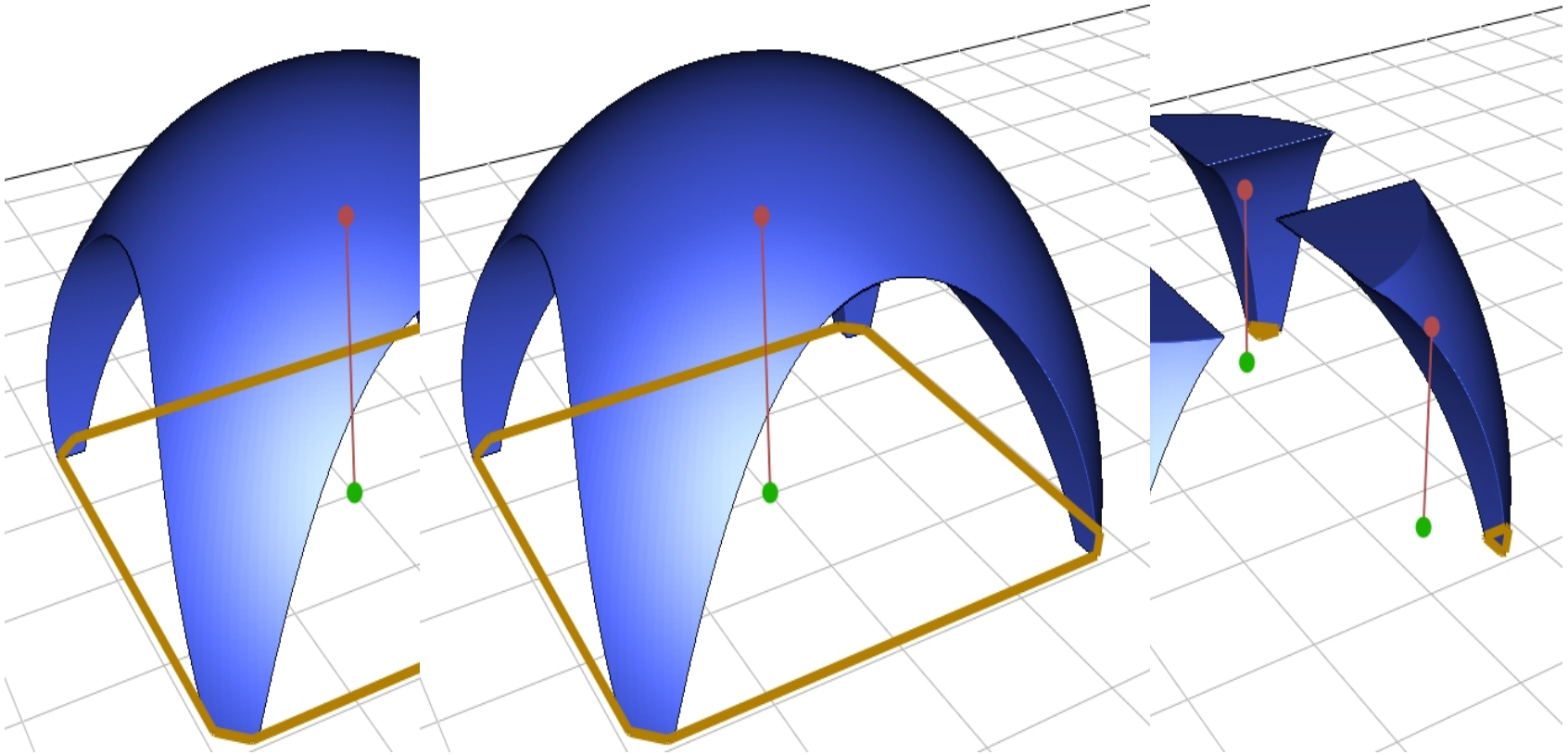




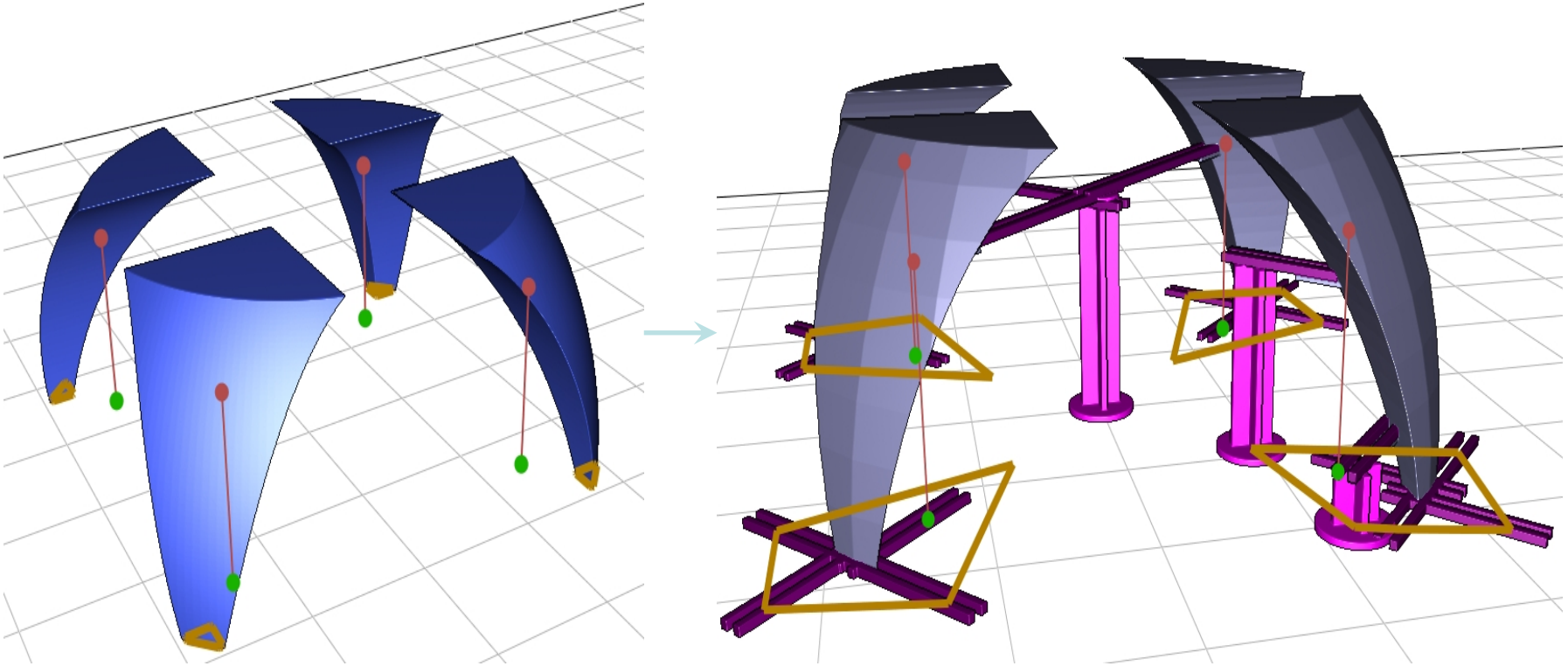
Algorithm Overview



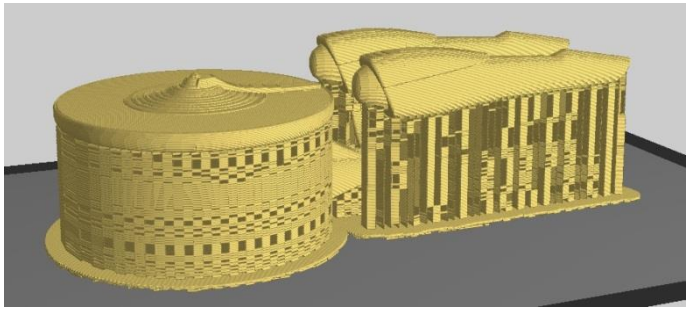
Stability During Printing



Stability Enhancement

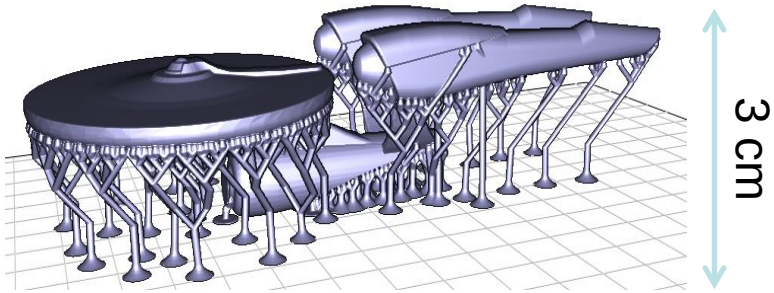


Comparison (Enterprise)



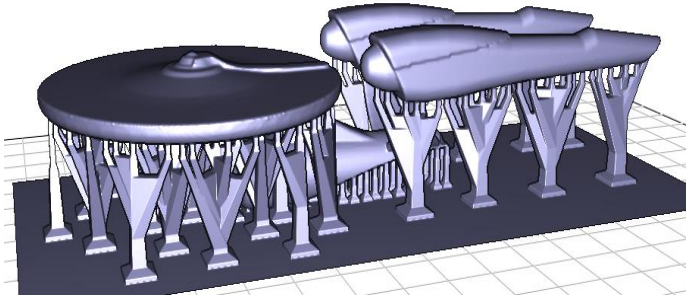
Makerware

11.9m 3h33



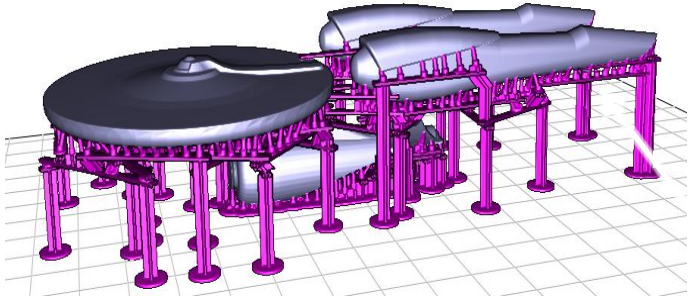
MeshMixer

2.5m 3h38



PhotoshopCC

4.61m N/A



New Method

2.69m 3h14

Results - Minotaur (New Method)

10 cm

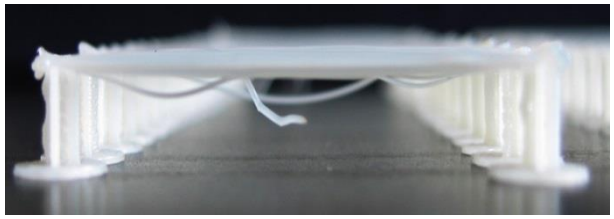
thing:46646

Results - Hilbert Cube

thing:16343

Limitations

- Sagging (see paper)
- Surface quality
- Computation time

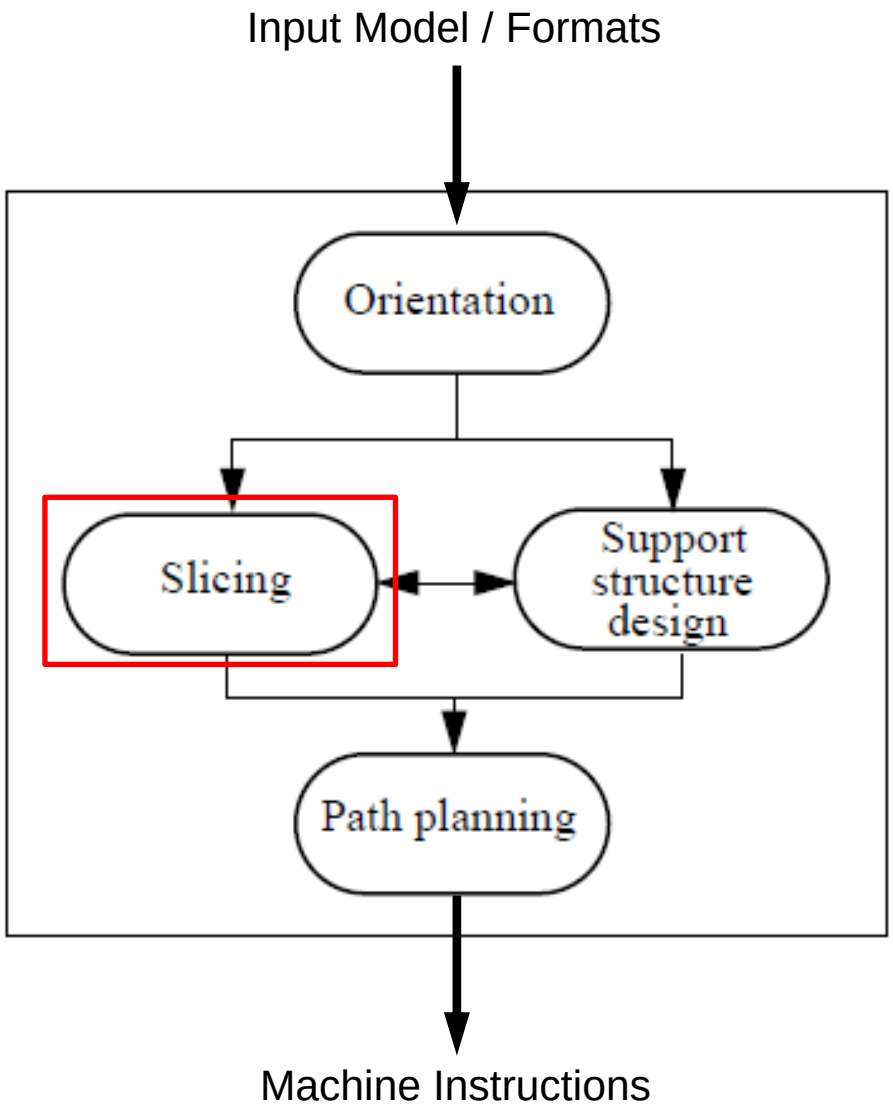


Support Structure Generation:

Ongoing Research Topic

Possible project topic!

3D Printing Process

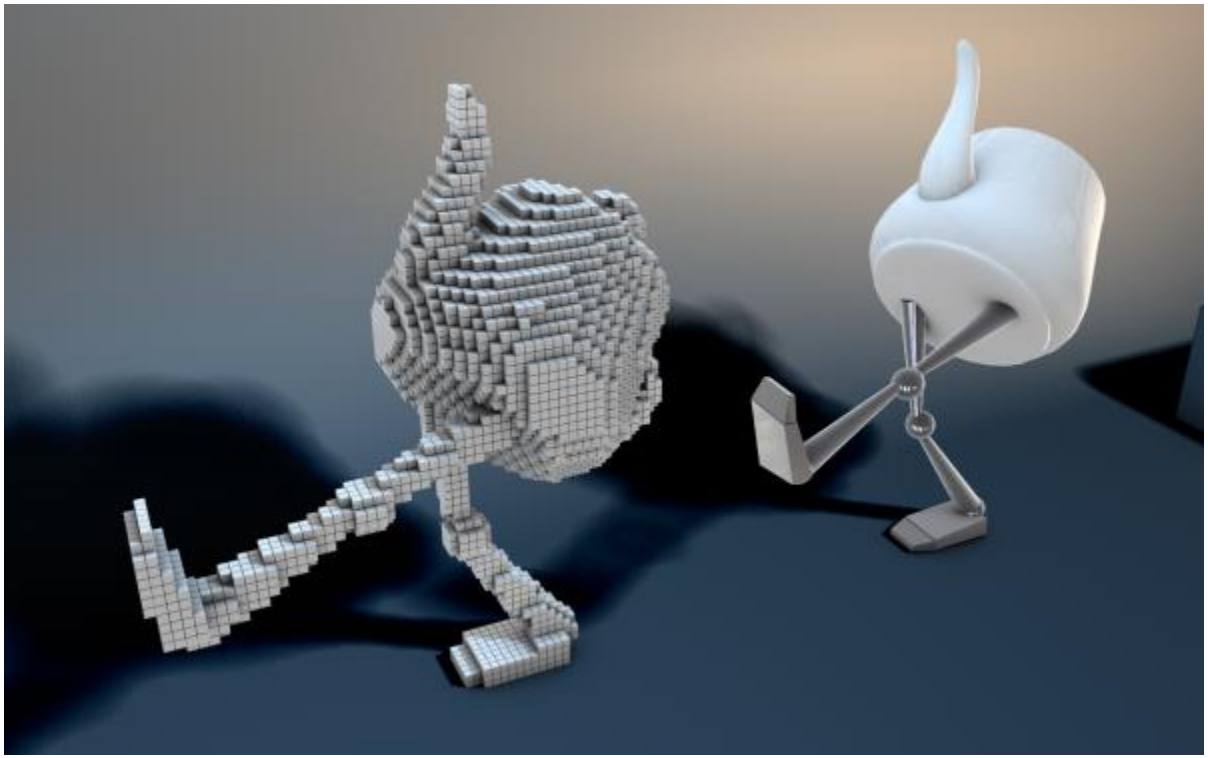


Slicing Demo

Quick NetFabb slicing demo

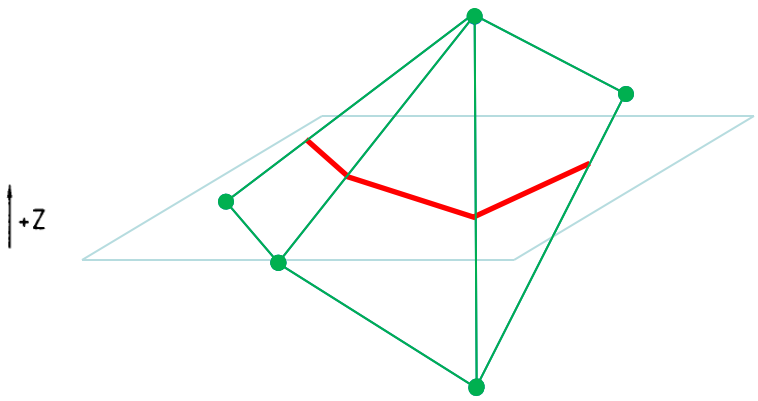
Slicing STL Models: Voxelization

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



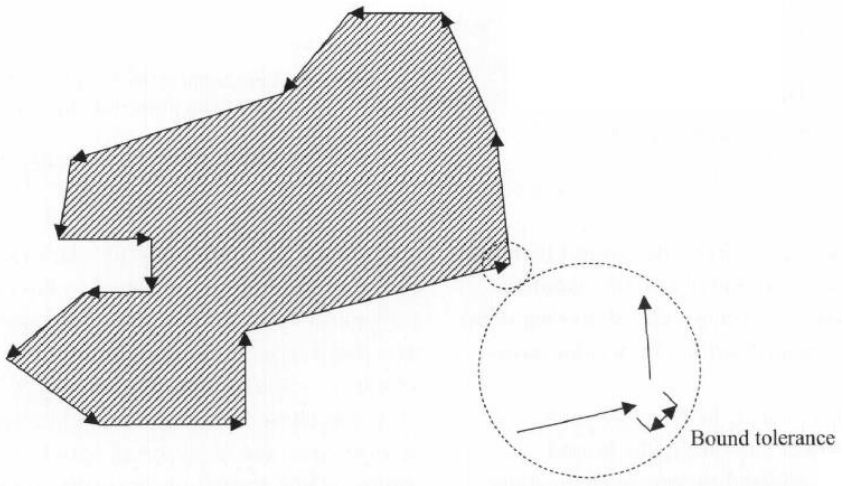
Slicing STL Models: direct approach

- 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



Slicing STL Models: direct approach

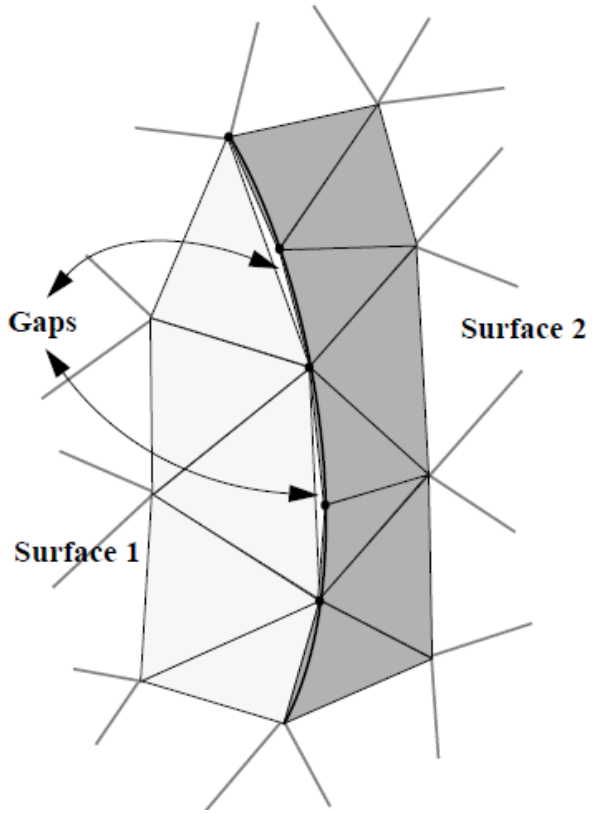
- STL does not store connectivity – “triangle soup”
- Algorithm:
  - For each z plane
    - For each triangle
      - Intersect triangle with the z plane
      - If they intersect, store line segment
    - Connect line segments, store contours



From Choi and Kwok, 2002

Slicing STL Models: Issues

- STL models are not always watertight -> epsilons

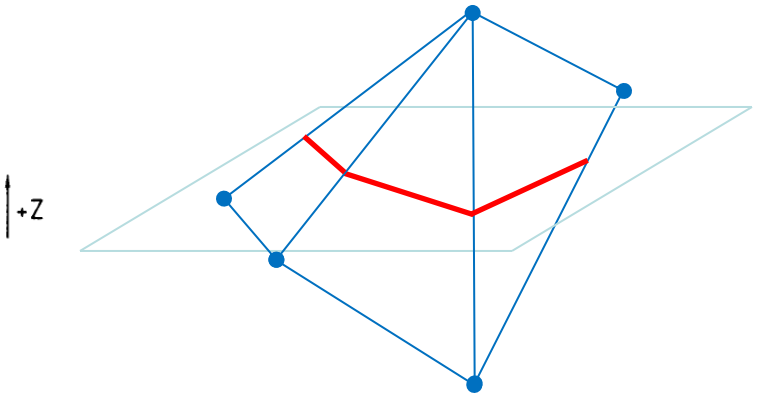


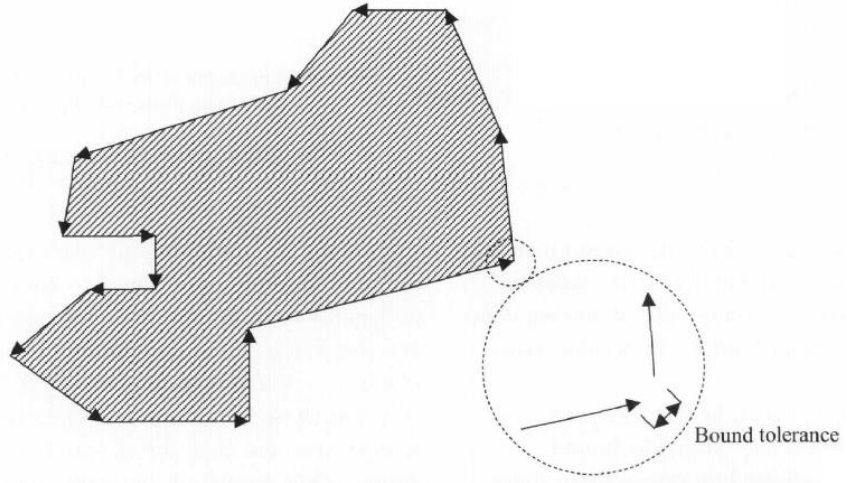
Marsan et al, 1998

Slicing STL Models: Issues

- 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

Very Slow!



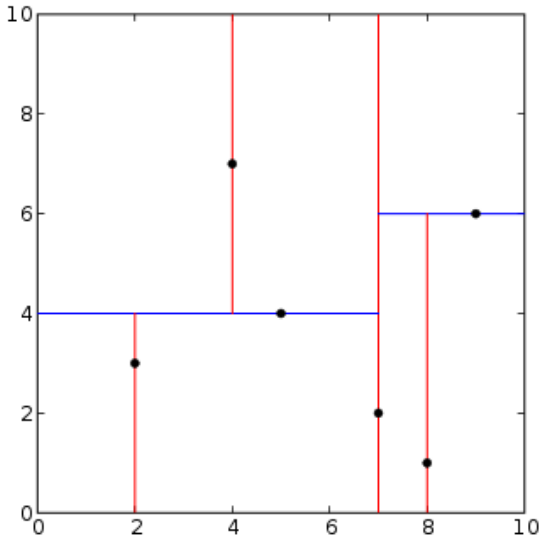


More Efficient Slicing

- 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**

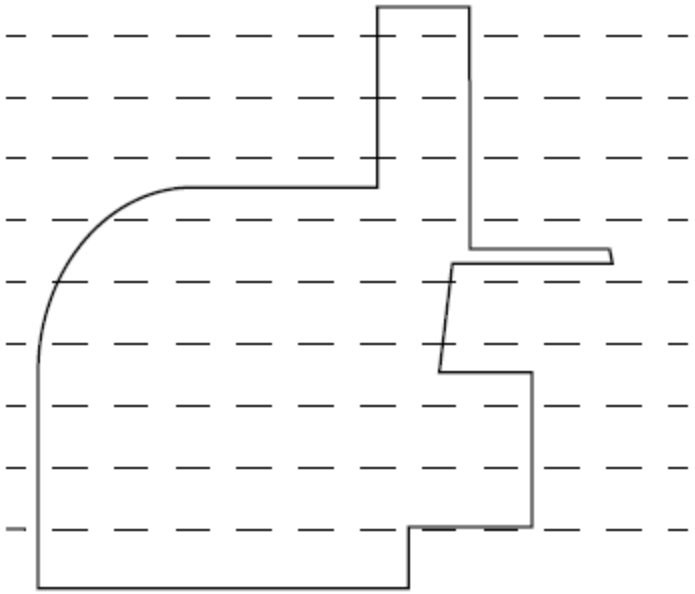
More Efficient Slicing

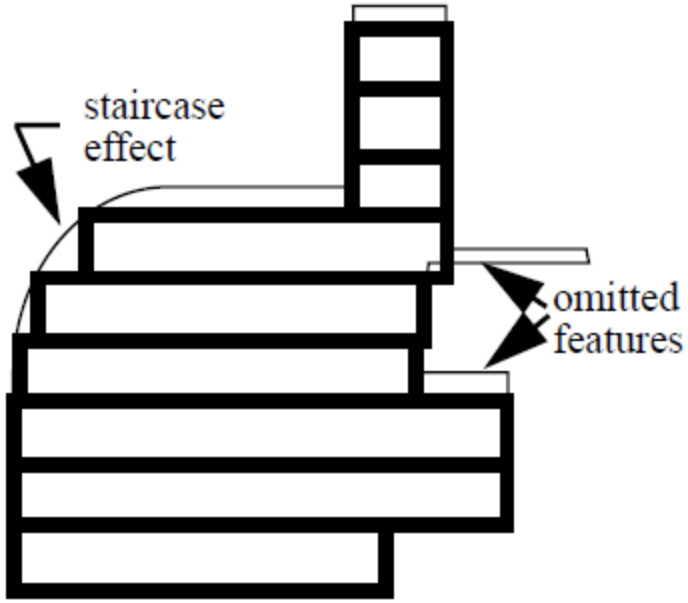
- Slicing is still a **bottleneck** when working with large models
- Many opportunities for very fast algorithms
  - Efficient out-of-core methods
    - when model does not fit in the memory
  - Using acceleration data structures
  - Z-sorting
  - GPU-based methods
- Possible project ideas!



Slicing Issues

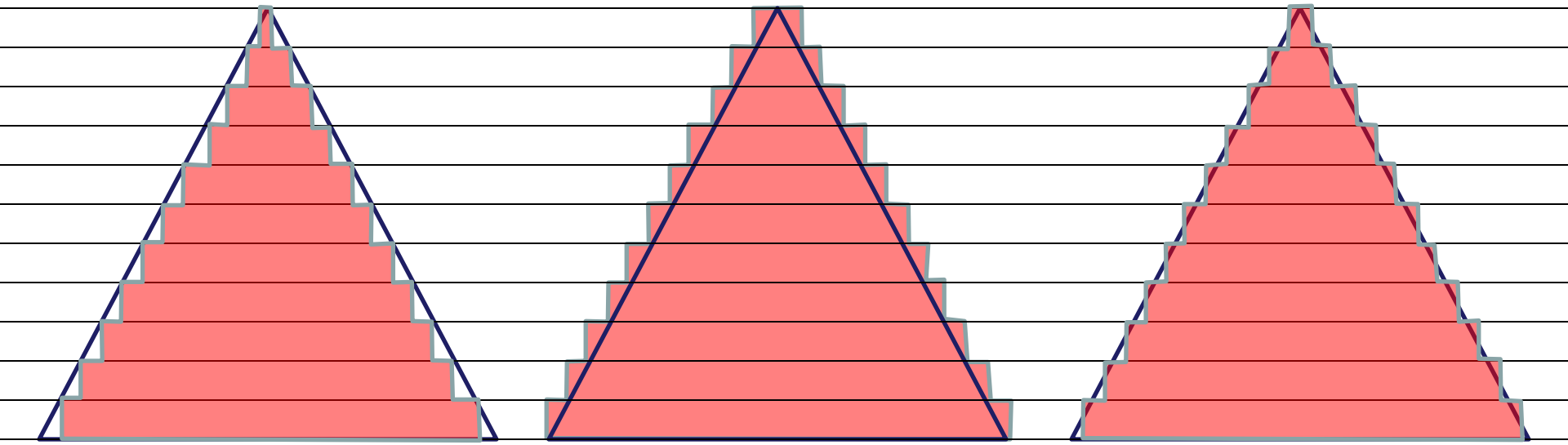
- Discretization == lossy





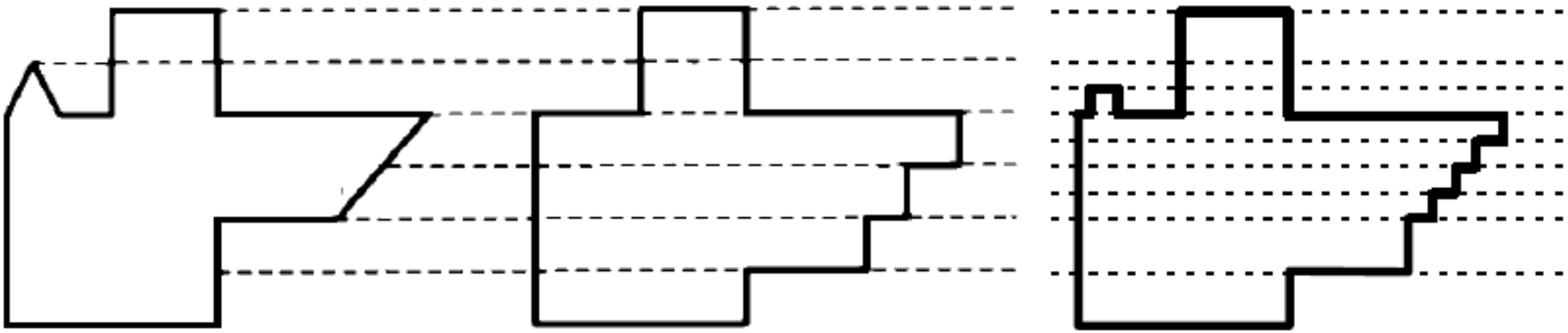
Slicing Issues

- 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



Adaptive Slicing

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



input model

uniform slicing

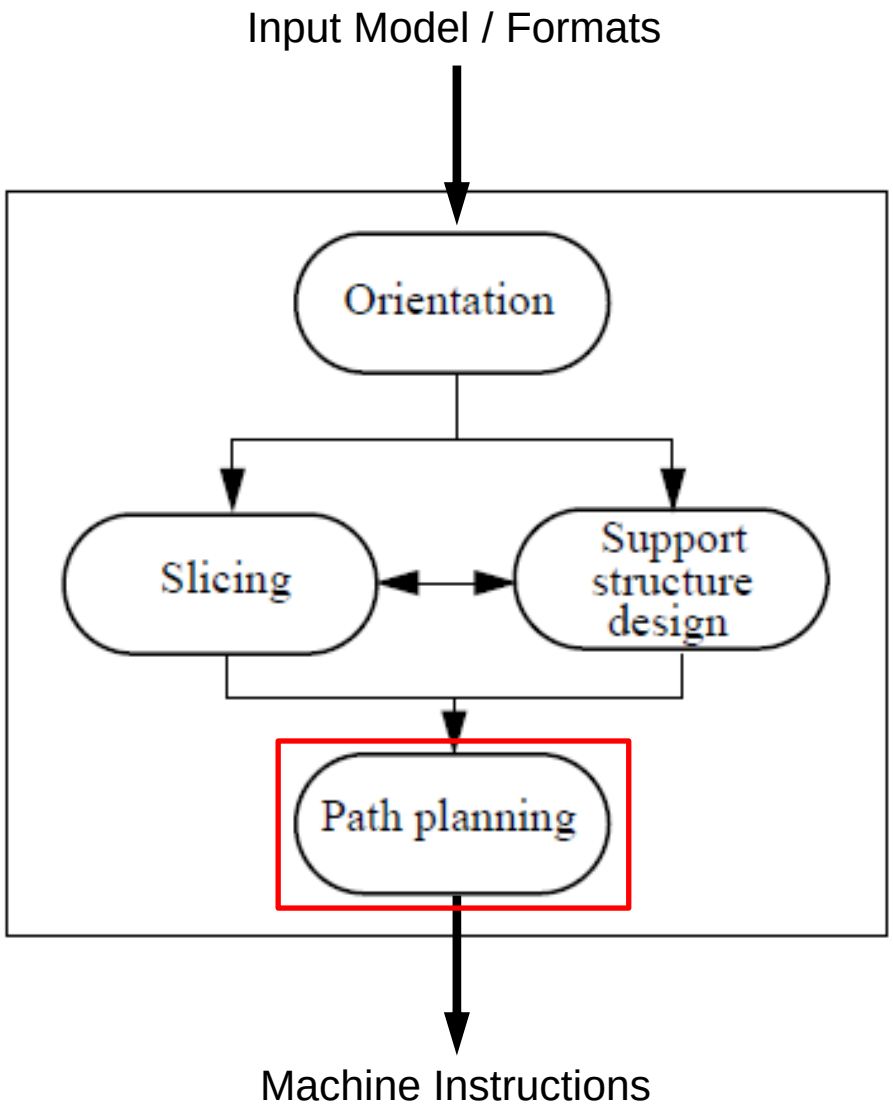
adaptive slicing

Slice File Formats

- Common Layer Interface (CLI) format
  - defined as polylines
  - both contour and hatch (fill pattern)
  - vendor independent format

- SLC file format by 3D Systems
  - defined as polylines
  - both contour and interior

3D Printing Process



Path Planning

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

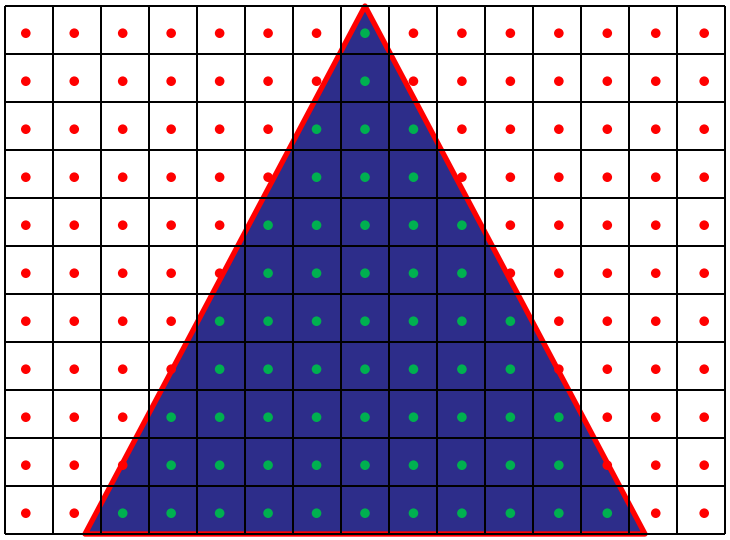
- Paths
  - Affect build time, surface accuracy, stiffness, strength, post-manufacture distortion

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

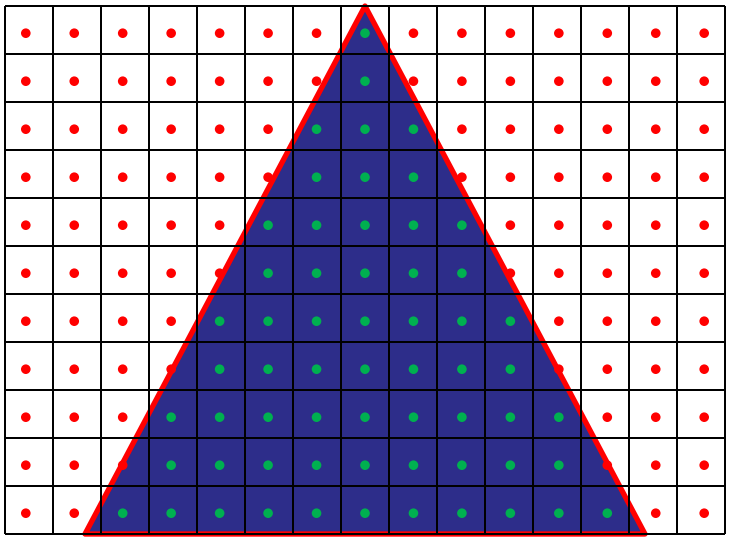
Simple Path Planning for Raster-based 3D Printing

- 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



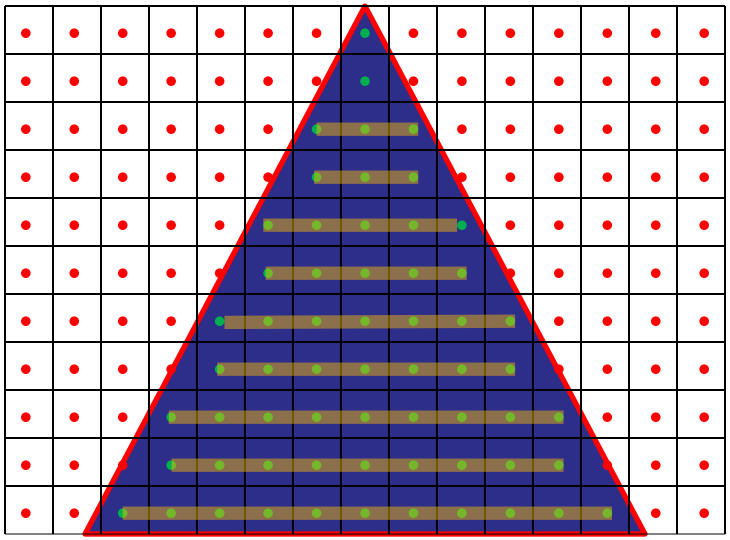
Simple Path Planning for Vector-based 3D Printing

- 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



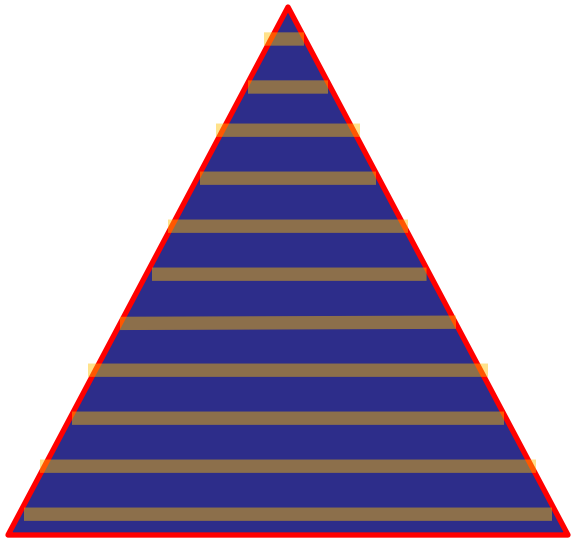
Simple Path Planning for Vector-based 3D Printing

- 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



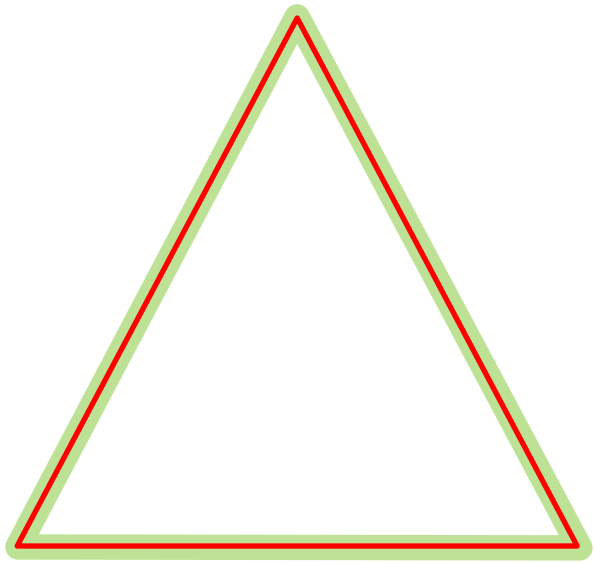
Simple Path Planning for Vector-based 3D Printing II

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

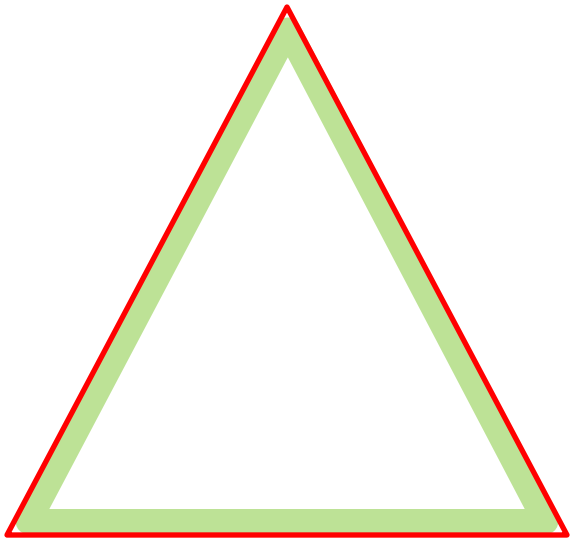


Tracing Contours

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



no offset



offset inwards

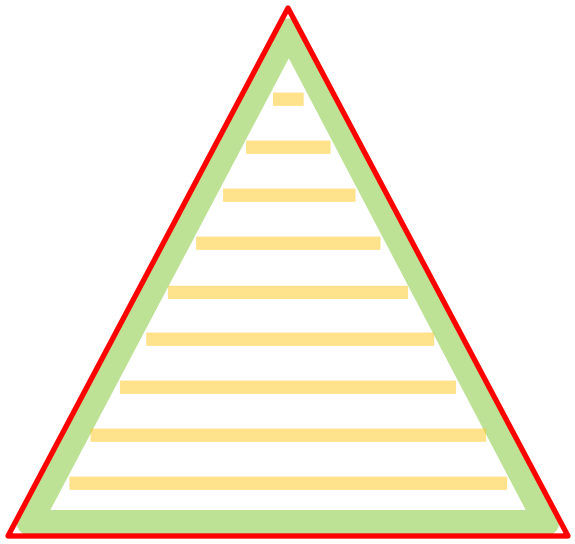
Tracing Contours

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

http://www.3ders.org

Tracing Contours

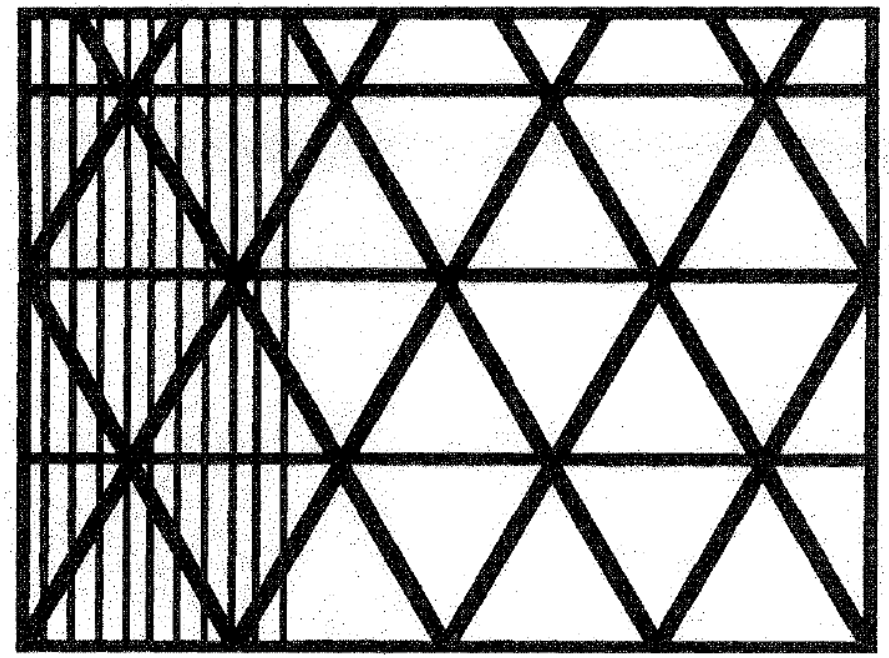
- Can be combined with filling the interior
- Interior fill paths do not extend from border to border
  - stopped short of the contour



contour offset inwards + interior fill

Advanced Fill Patterns

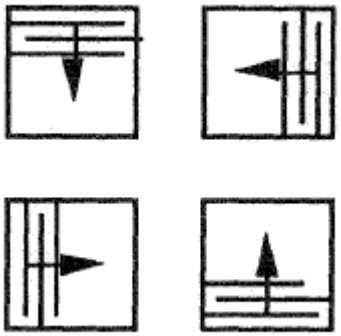
- TriHatch (developed by 3D Systems)
  - developed for stereolithography (SLA)
  - interior layer is filled with equilateral triangles
  - skins fills on the bottom and top of the part



Horton et al 1993

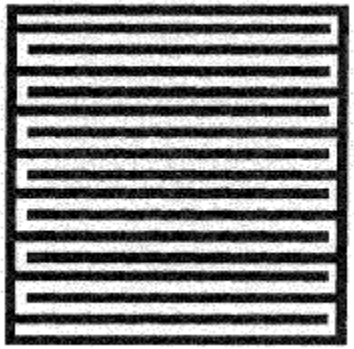
Advanced Fill Patterns

- 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

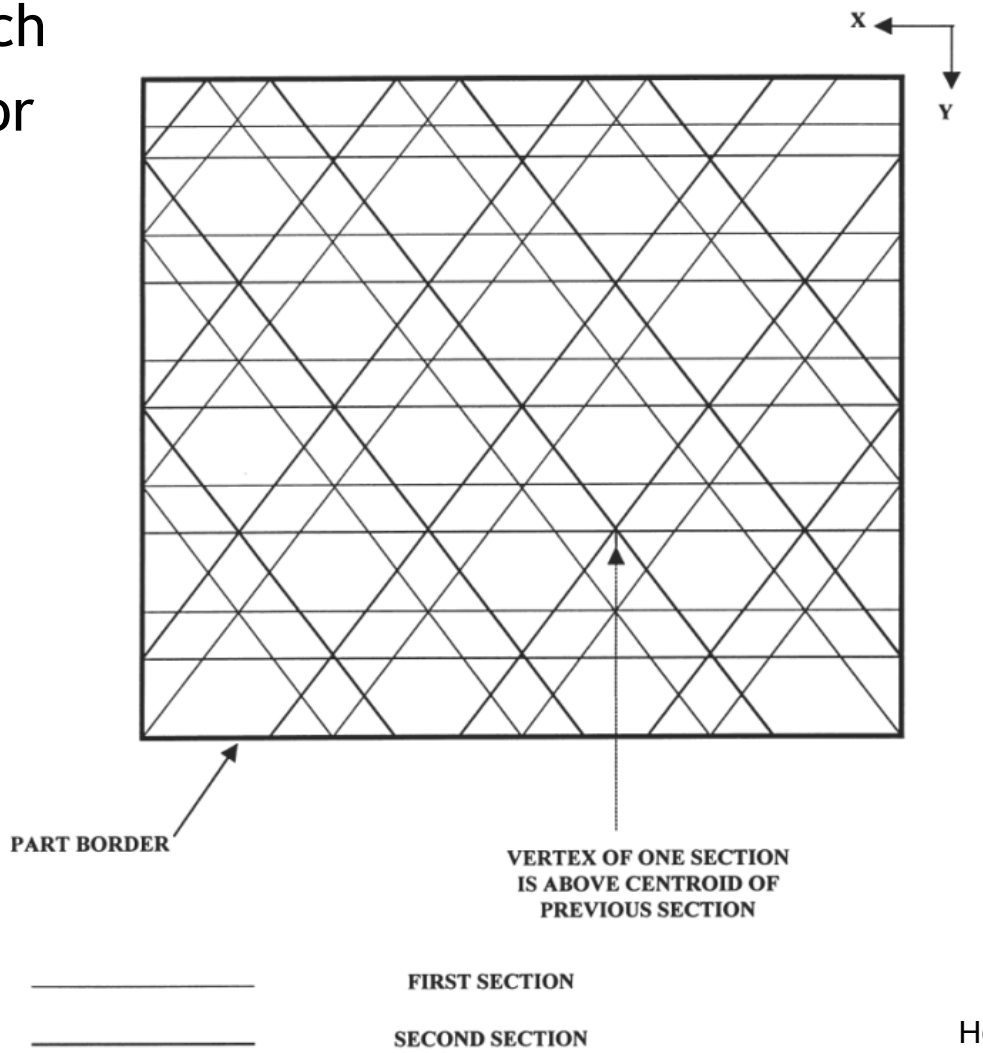


retracted hatch

Horton et al 1993

More Fill Patterns

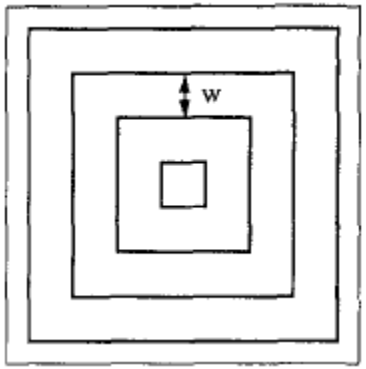
- QuickCast
  - Similar to TriHatch
  - Patterns offset for each layer



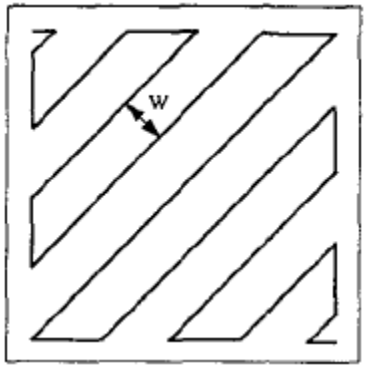
Horton et al 1993

More Fill Patterns

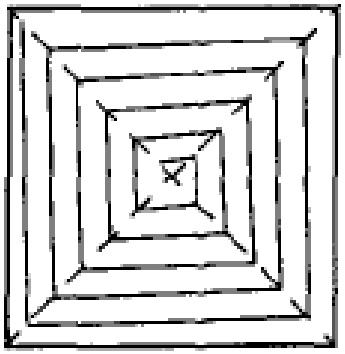
- Criteria: print speed, structural properties, weight vs strength, etc



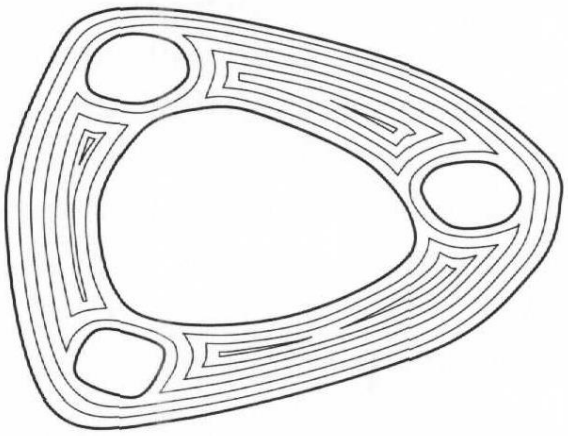
contour



raster



spiral

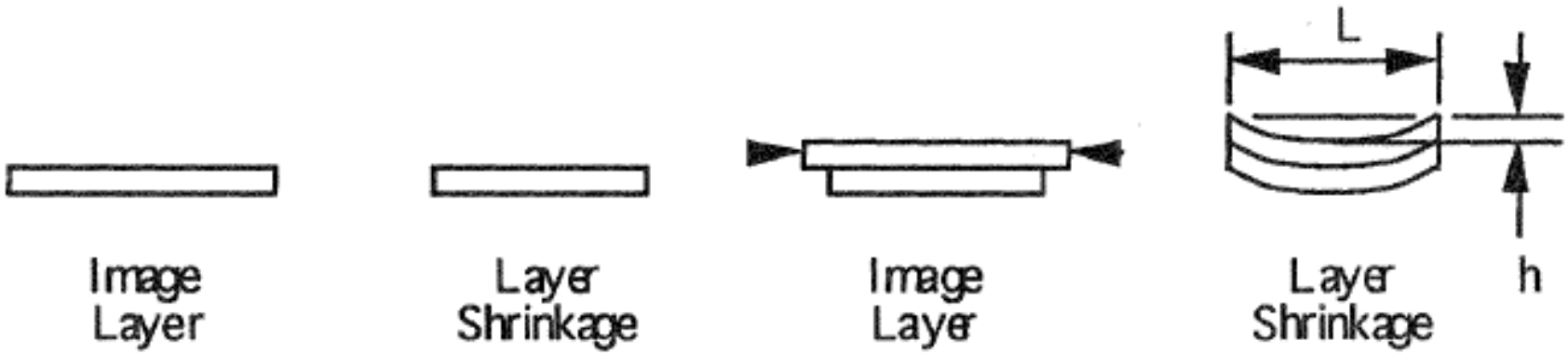


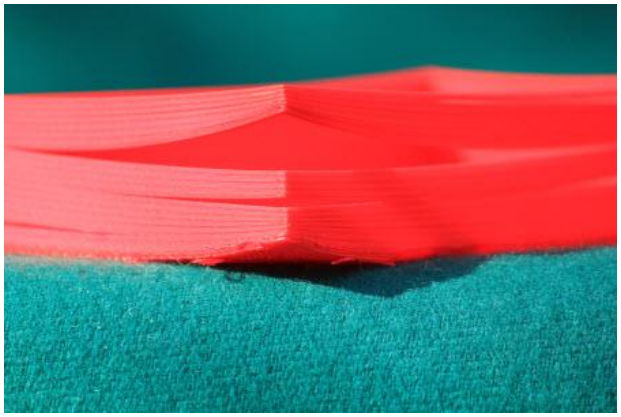
Project opportunity

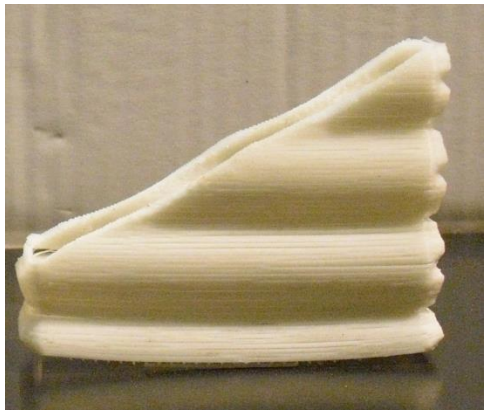
Kulkarni and Dutta 1999

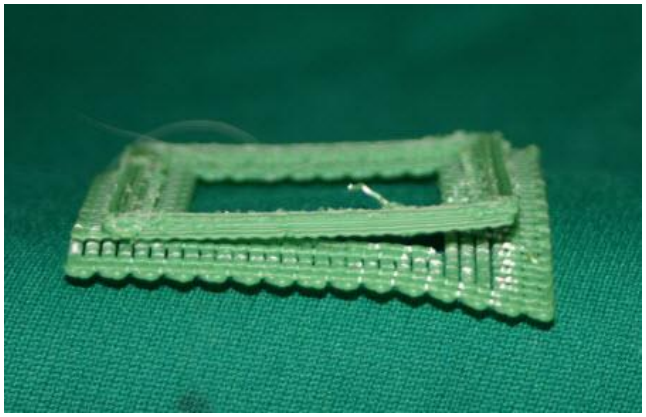
Material Shrinkage/Warping

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







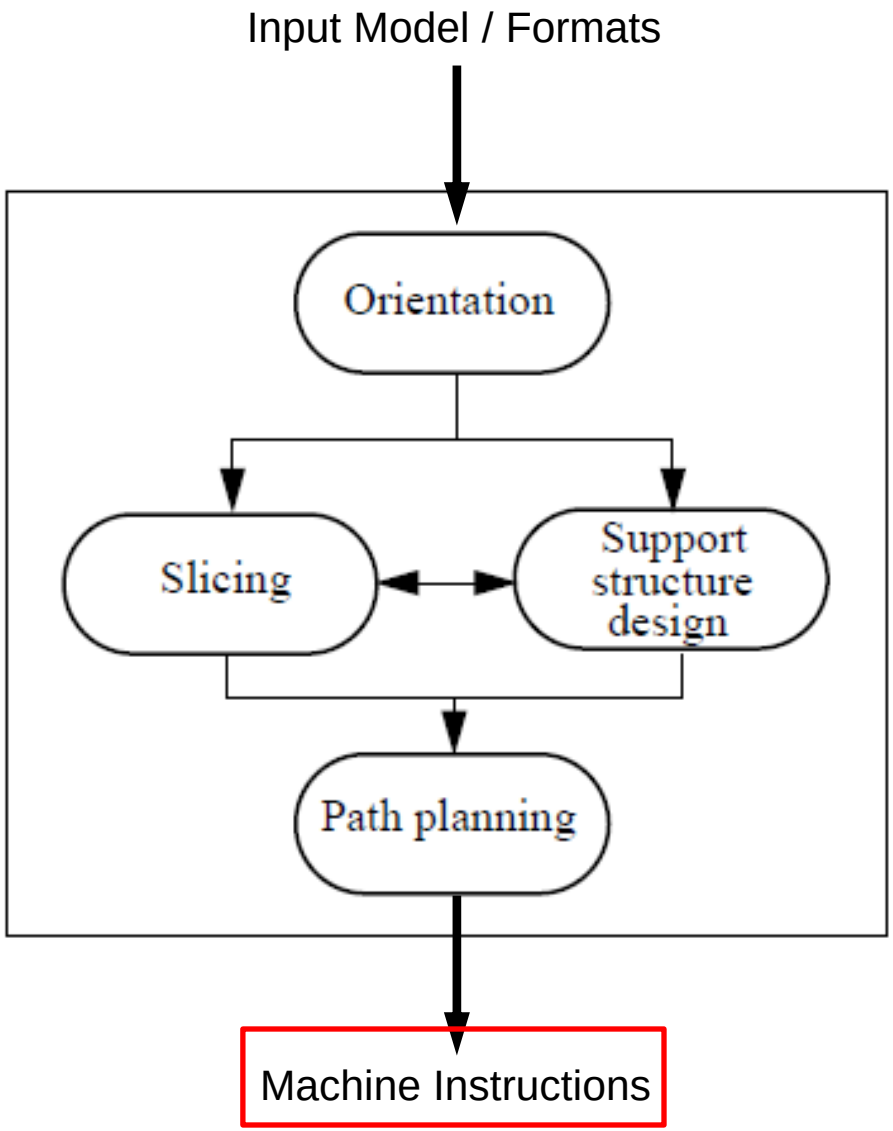


Source: http://cc-products.de, http://www.deelip.com

Path Planning Summary

- Minimal input is the slice description
- Optimal strategy requires knowledge of particular 3D Printing process, materials, etc

3D Printing Process



Machine Instructions

- Raster file formats
  - DLP 3D printing, plaster-based 3D printing, phase-change inkjets
  - Proprietary, not exposed
  - Can be exported as image files (e.g., PNG, BMP)
- Vector file formats
  - G-Code
  - SLI by 3D Systems – machine-specific 2D format for the vector 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.

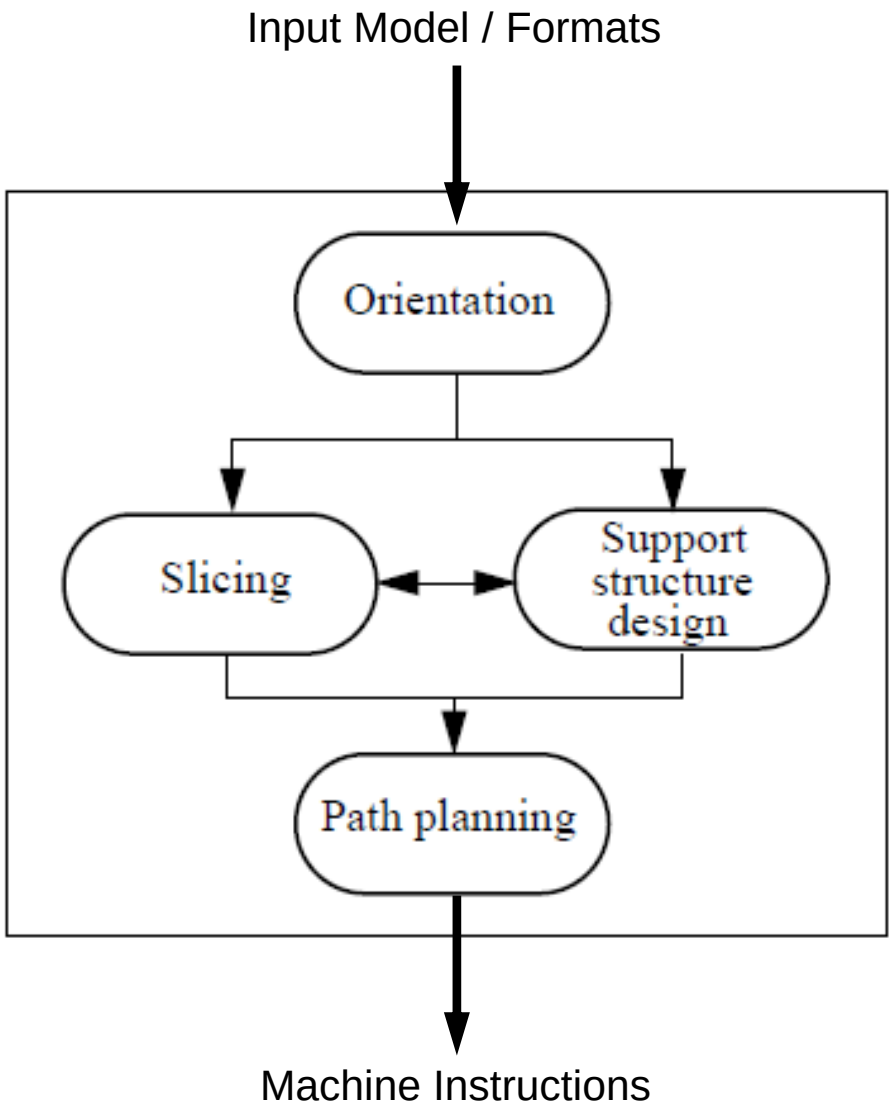
seek the Z-axis to 0.25“
travel to X=-0.5 and Y=0.0

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

3D Printing Process



Questions?

That’s All for Today!