Computational Fabrication

Wojciech Matusik, Adriana Schulz

SIGGRAPH '19 Courses, July 28 - August 01, 2019, Los Angeles, CA, USA
ACM 978-1-4503-6307-5/19/07.
10.1145/3305366.3328065

Your lecturers for this course

- Instructors
  - Wojciech Matusik, MIT
  - Adriana Schulz, UW

Course Website

www.yellkey.com/agent

Course Schedule

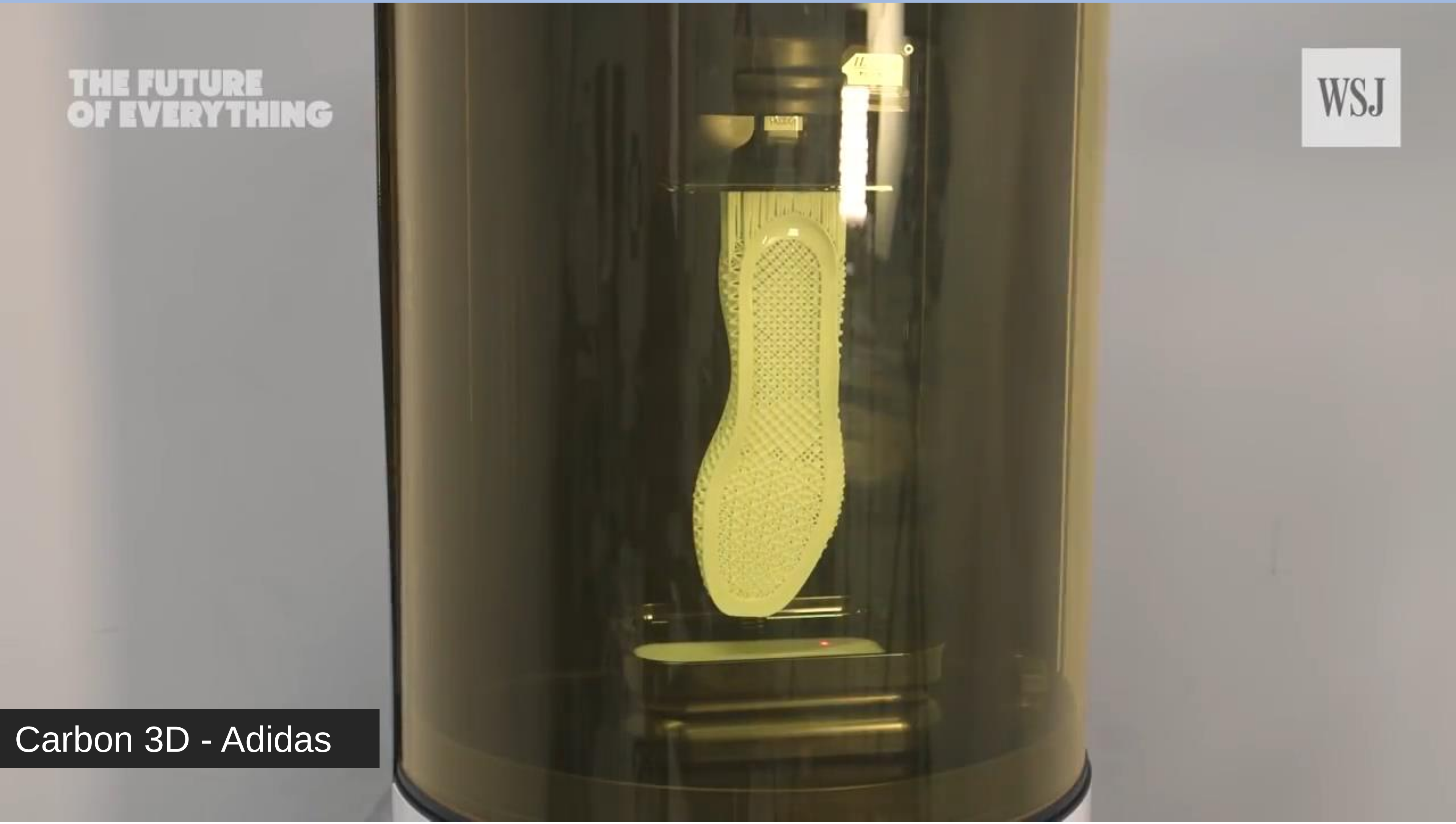
9:00 am – 9:10 am	Introduction
9:10 am – 9:25 am	Hardware Review
9:25 am – 9:50 am	From Design to Machine Code
9:50 am – 10:20 am	Design Space Representations
10:20 am – 10:25 am	Performance Driven Design
10:25 am - 10:40 am	Break
10:40 am – 10:55 am	Performance Space Representation
10:55 am – 11:15 am	Inverse Methods
11:15 am – 11:35 am	Multi-objective Inverse Methods
11:35 am – 12:00 pm	Advanced Performance Driven Design
12:00 pm – 12:15 pm	Course Review

Course Schedule

9:00 am – 9:10 am	Introduction
9:10 am – 9:25 am	Hardware Review
9:25 am – 9:50 am	From Design to Machine Code
9:50 am – 10:20 am	Design Space Representations
10:25 am – 10:25 am	Performance Driven Design
10:25 am - 10:40 am	Break
10:40 am – 10:55 am	Performance Space Representation
10:55 am - 11:15 am	Inverse Methods
11:15 am - 11:35 am	Multi-objective Inverse Methods
11:35 am - 12:00 pm	Advanced Performance Driven Design
12:00 pm - 12:15 pm	Course Review

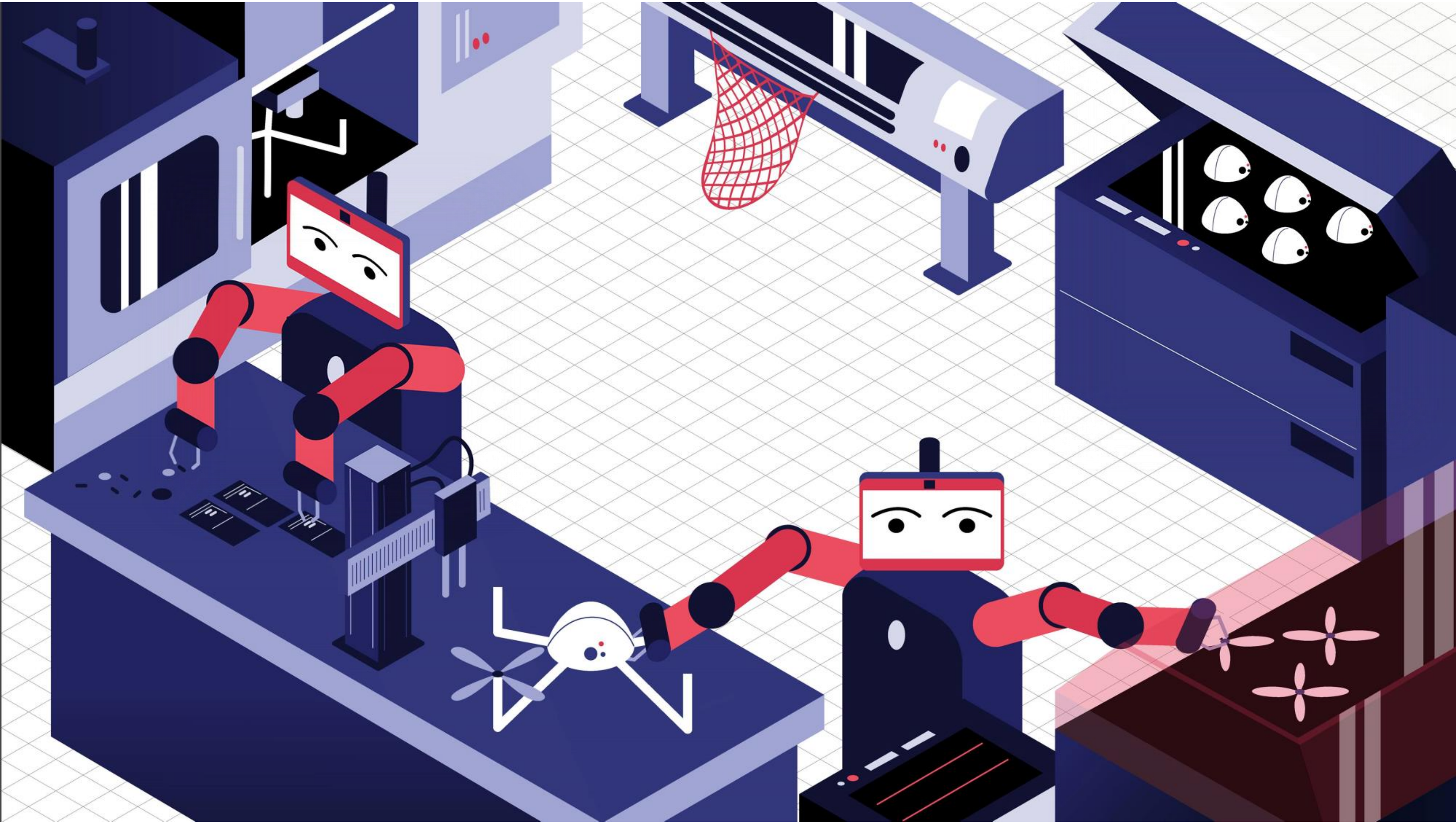
Manufacturing Today

Manufacturing in Near Future



Carbon 3D - Adidas

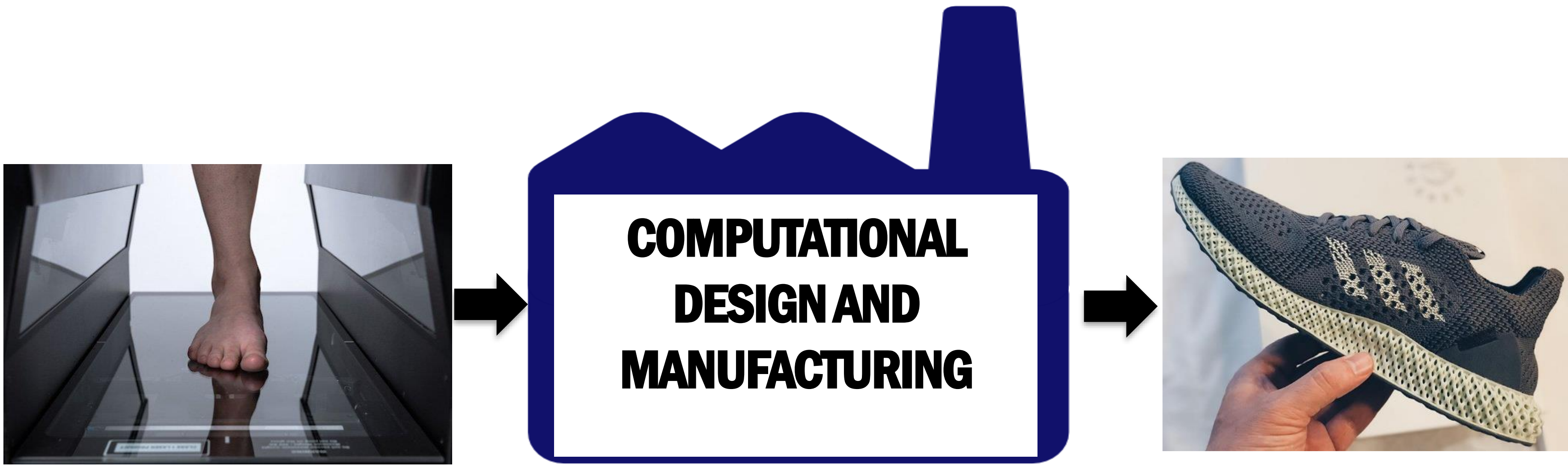
Manufacturing in Near Future



Manufacturing in Near Future

Why Computational Design and Manufacturing?

Why Computational Design and Manufacturing?



Hardware

Hardware

Hardware Abstraction & Machine Code


```
;Layer count: 336
;LAYER:0
M107
G0 F9000 X91.800 Y93.520 Z0.300
;TYPE:SKIRT
G1 F1200 X92.617 Y92.870 E0.01964
G1 X93.518 Y92.412 E0.03865
G1 X94.458 Y92.141 E0.05705
G1 X95.218 Y92.072 E0.07141
G1 X95.998 Y92.064 E0.08608
G1 X96.894 Y92.071 E0.10294
G1 X98.900 Y92.070 E0.14067
```

Hardware

Machine Code

Hardware Abstraction & Machine Code

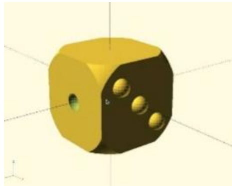
```
;Layer count: 336
;LAYER:0
M107
G0 F9000 X91.800 Y93.520 Z0.300
;TYPE:SKIRT
G1 F1200 X92.617 Y92.870 E0.01964
G1 X93.518 Y92.412 E0.03865
G1 X94.458 Y92.141 E0.05705
G1 X95.218 Y92.072 E0.07141
G1 X95.998 Y92.064 E0.08608
G1 X96.894 Y92.071 E0.10294
G1 X98.900 Y92.070 E0.14067
```

Hardware

Machine Code

Design: Shape and Materials


```
;Layer count: 336
;LAYER:0
M107
G0 F9000 X91.800 Y93.520 Z0.300
;TYPE:SKIRT
G1 F1200 X92.617 Y92.870 E0.01964
G1 X93.518 Y92.412 E0.03865
G1 X94.458 Y92.141 E0.05705
G1 X95.218 Y92.072 E0.07141
G1 X95.998 Y92.064 E0.08608
G1 X96.894 Y92.071 E0.10294
G1 X98.900 Y92.070 E0.14067
```



```
difference(){
//cuerpo del dado
intersection(){
cube(20,center=true);
sphere(15,$fn=100);}
//cara del 1
translate([10,0,0])
sphere(2,$fn=20);
}
translate([0,10,0])
sphere(2,$fn=20);
translate([5,10,5])
sphere(2,$fn=20);
translate([-5,10,-5])
sphere(2,$fn=20);
```

Hardware

Machine Code

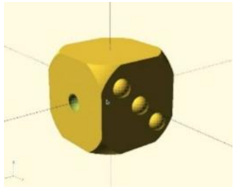
Design

From Design to Machine Code

Hardware

```
;Layer count: 336
;LAYER:0
M107
G0 F9000 X91.800 Y93.520 Z0.300
;TYPE:SKIRT
G1 F1200 X92.617 Y92.870 E0.01964
G1 X93.518 Y92.412 E0.03865
G1 X94.458 Y92.141 E0.05705
G1 X95.218 Y92.072 E0.07141
G1 X95.998 Y92.064 E0.08608
G1 X96.894 Y92.071 E0.10294
G1 X98.900 Y92.070 E0.14067
```

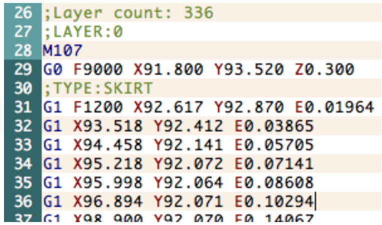
Machine Code



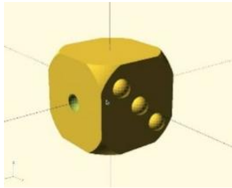
```
difference(){
//cuerpo del dado
intersection(){
cube(20,center=true);
sphere(15,$fn=100);}
//cara del 1
translate([10,0,0])
sphere(2,$fn=20);
}
translate([0,10,0])
sphere(2,$fn=20);
translate([5,10,5])
sphere(2,$fn=20);
translate([-5,10,-5])
sphere(2,$fn=20);
```

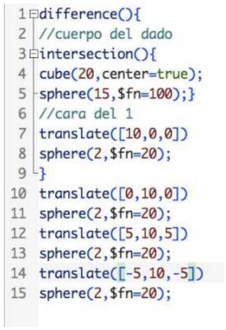
Design

High-level Specifications/Performance

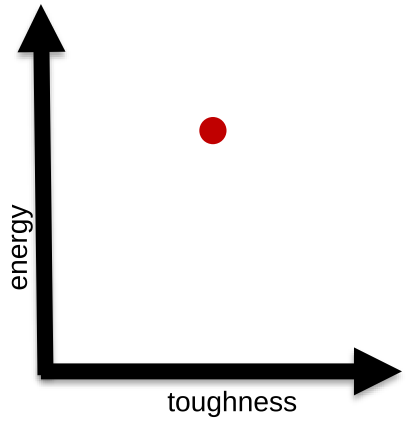


```
;Layer count: 336
;LAYER:0
M107
G0 F9000 X91.800 Y93.520 Z0.300
;TYPE:SKIRT
G1 F1200 X92.617 Y92.870 E0.01964
G1 X93.518 Y92.412 E0.03865
G1 X94.458 Y92.141 E0.05705
G1 X95.218 Y92.072 E0.07141
G1 X95.998 Y92.064 E0.08608
G1 X96.894 Y92.071 E0.10294
```





```
difference(){
//cuerpo del dado
intersection(){
cube(20,center=true);
sphere(15,$fn=100);}
//cara del 1
translate([10,0,0])
sphere(2,$fn=20);
}
translate([0,10,0])
sphere(2,$fn=20);
translate([5,10,5])
sphere(2,$fn=20);
translate([-5,10,-5])
sphere(2,$fn=20);
```



Hardware

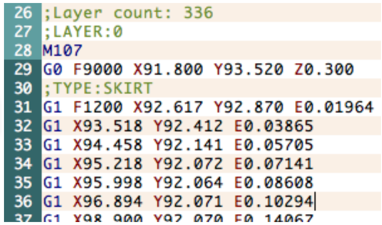
Machine Code

Design

Performance

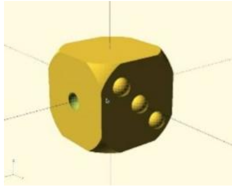
From Design to Performance

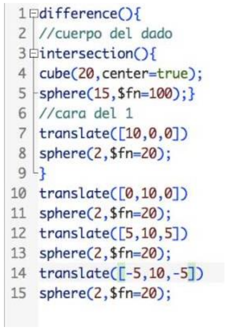
Hardware



```
;Layer count: 336
;LAYER:0
M107
G0 F9000 X91.800 Y93.520 Z0.300
;TYPE:SKIRT
G1 F1200 X92.617 Y92.870 E0.01964
G1 X93.518 Y92.412 E0.03865
G1 X94.458 Y92.141 E0.05705
G1 X95.218 Y92.072 E0.07141
G1 X95.998 Y92.064 E0.08608
G1 X96.894 Y92.071 E0.10294
```

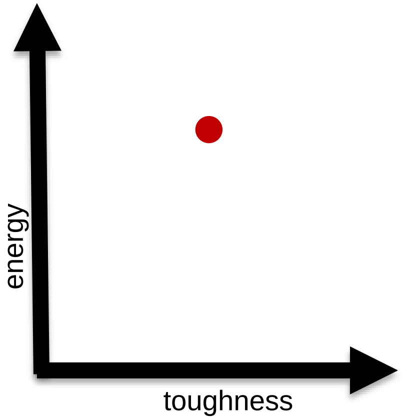
Machine Code





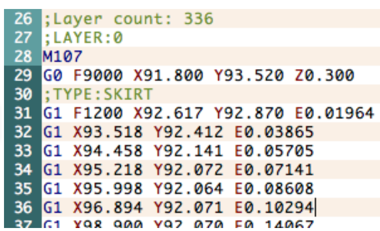
```
difference(){
//cuerpo del dado
intersection(){
cube(20,center=true);
sphere(15,$fn=100);}
//cara del 1
translate([10,0,0])
sphere(2,$fn=20);
}
translate([0,10,0])
sphere(2,$fn=20);
translate([5,10,5])
sphere(2,$fn=20);
translate([-5,10,-5])
sphere(2,$fn=20);
```

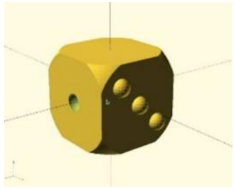
Design

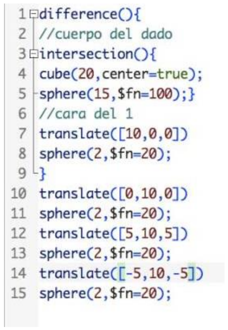


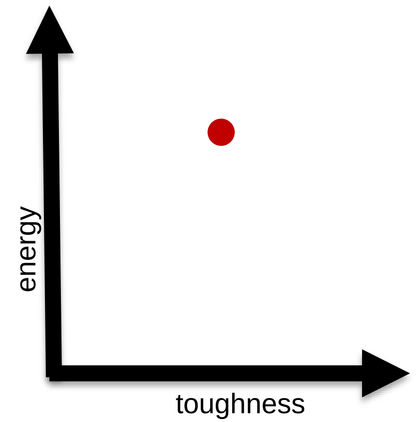
Performance

From Performance to Design









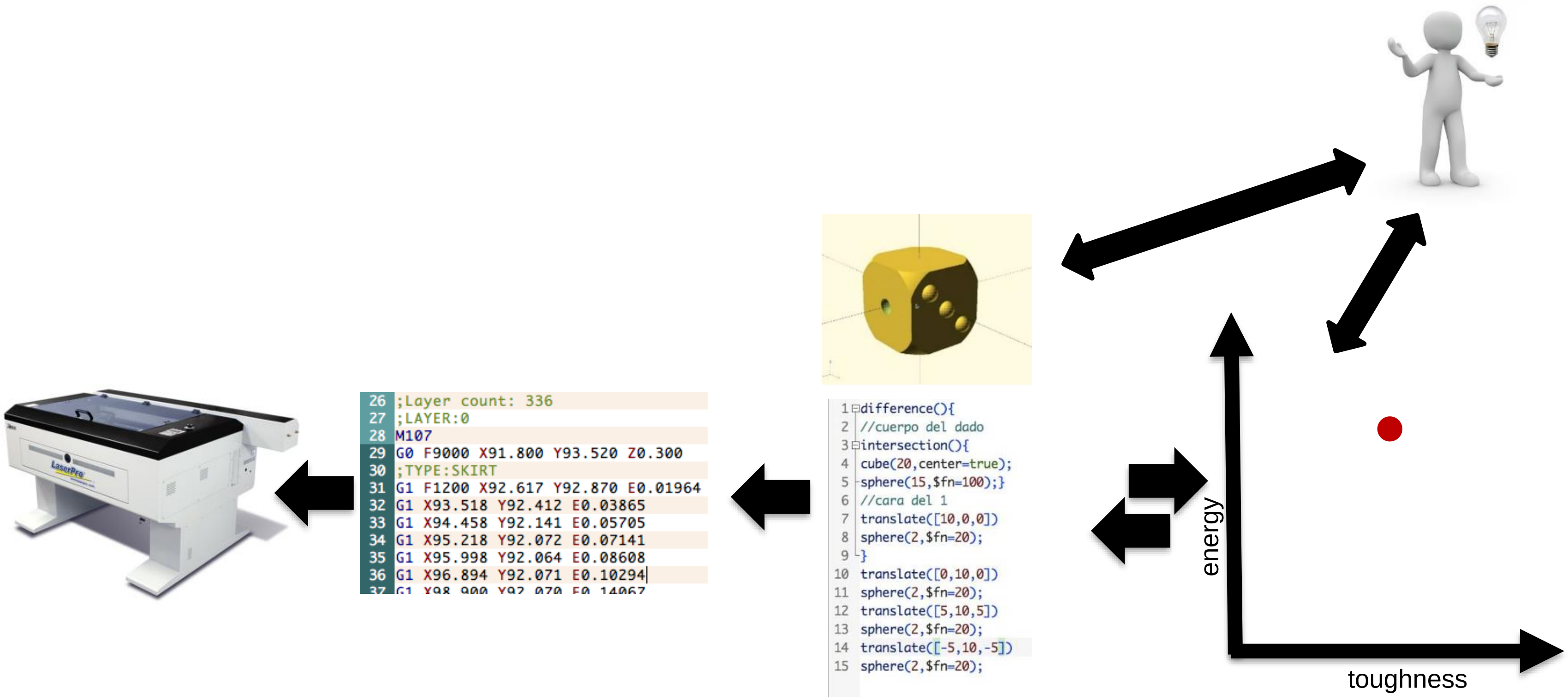
Hardware

Machine Code

Design

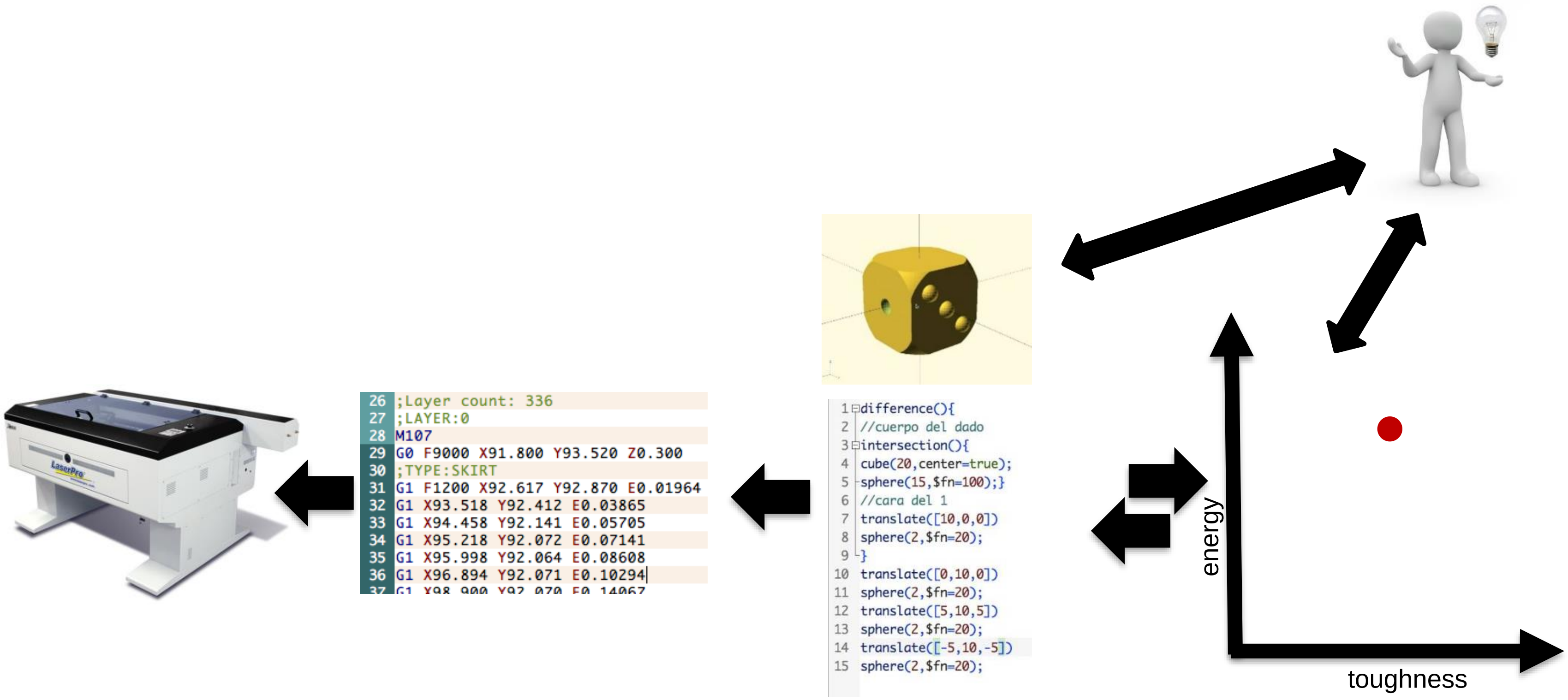
Performance

Human Computer Interaction (HCI)



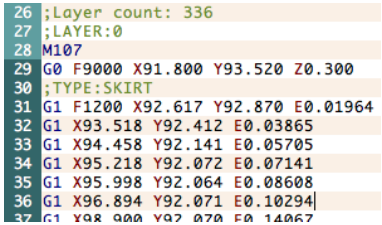
Hardware | **Machine Code** | **Design** | **Performance**

Artificial Intelligence (AI) & Machine Learning

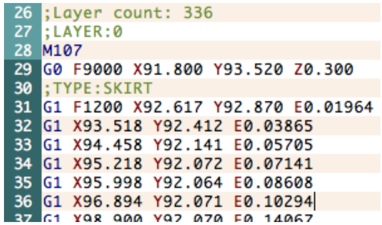


Hardware | **Machine Code** | **Design** | **Performance**

Operating Systems for Future Factory



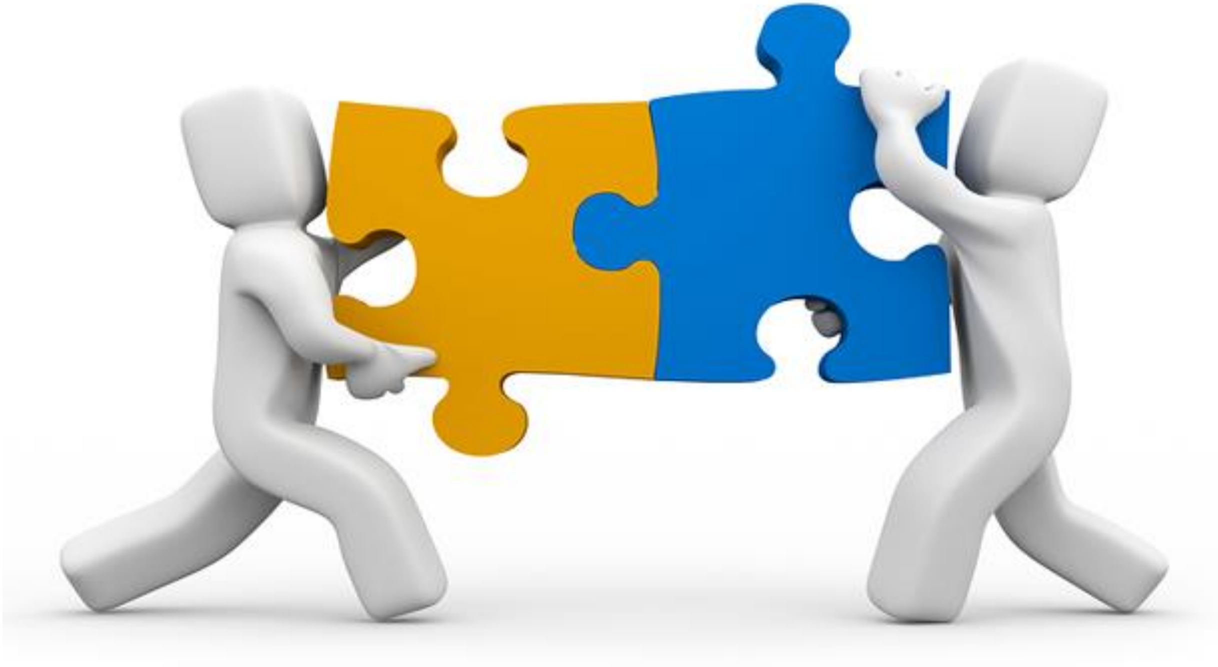
```
;Layer count: 336
;LAYER:0
M107
G0 F9000 X91.800 Y93.520 Z0.300
;TYPE:SKIRT
G1 F1200 X92.617 Y92.870 E0.01964
G1 X93.518 Y92.412 E0.03865
G1 X94.458 Y92.141 E0.05705
G1 X95.218 Y92.072 E0.07141
G1 X95.998 Y92.064 E0.08608
G1 X96.894 Y92.071 E0.10294
```



```
;Layer count: 336
;LAYER:0
M107
G0 F9000 X91.800 Y93.520 Z0.300
;TYPE:SKIRT
G1 F1200 X92.617 Y92.870 E0.01964
G1 X93.518 Y92.412 E0.03865
G1 X94.458 Y92.141 E0.05705
G1 X95.218 Y92.072 E0.07141
G1 X95.998 Y92.064 E0.08608
G1 X96.894 Y92.071 E0.10294
```

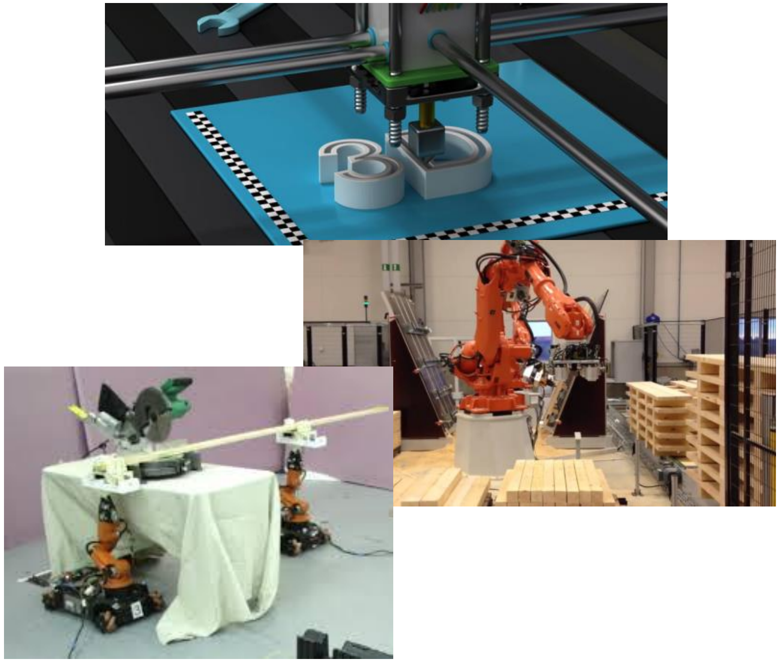
- Multi-tasking
- Distributed systems

Who's working on this?

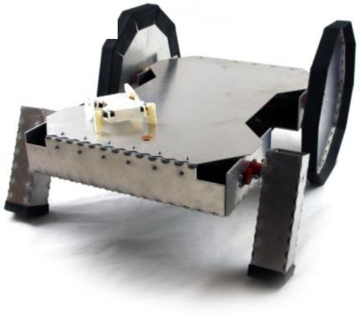


Highly interdisciplinary field!

Robotics

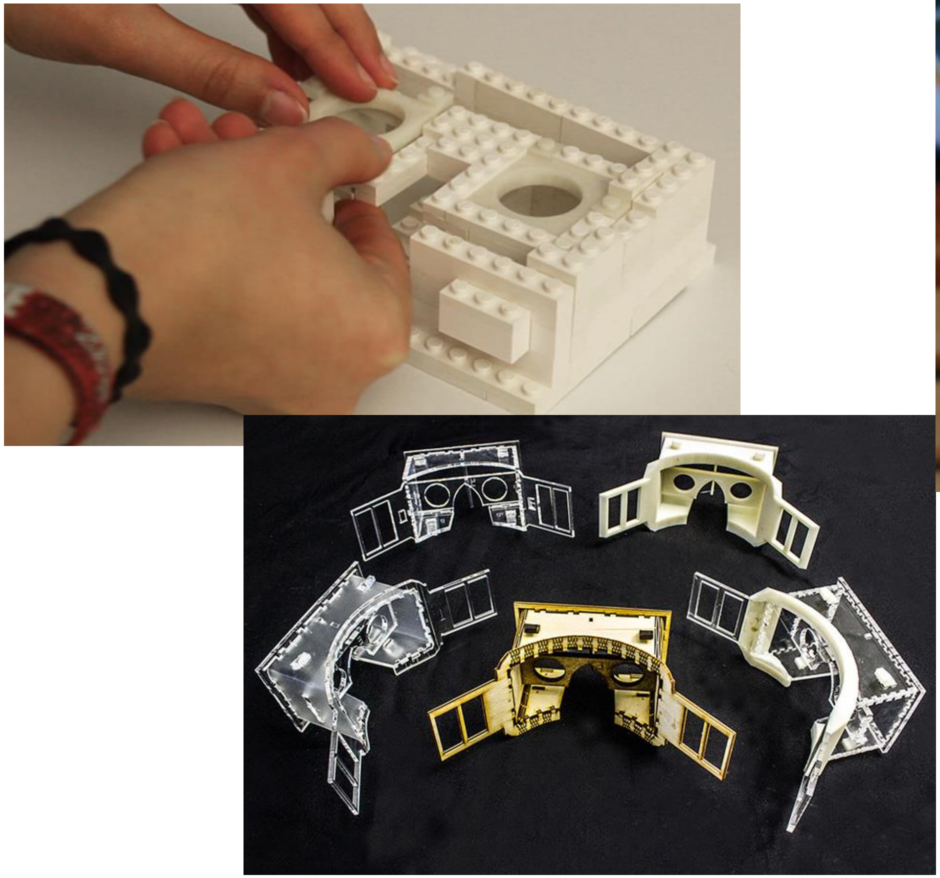


Fabrication Hardware



Design of Cyber-physical Systems

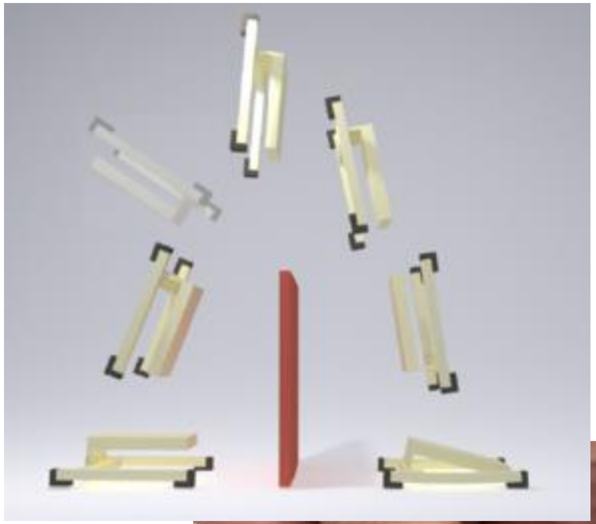
Human-Computer Interaction

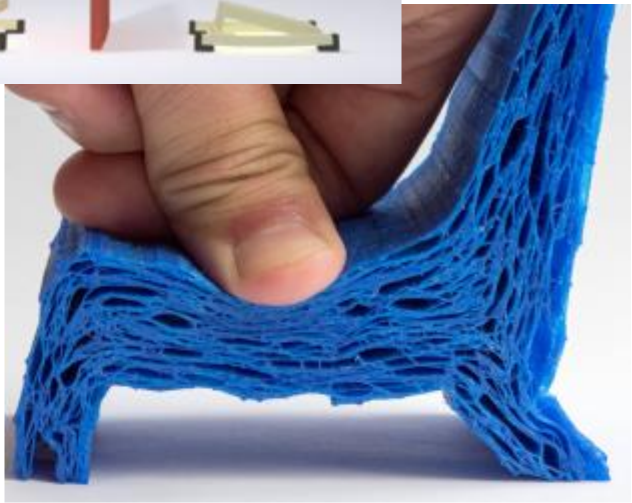


Hardware interfaces

Design Systems

Computer Graphics

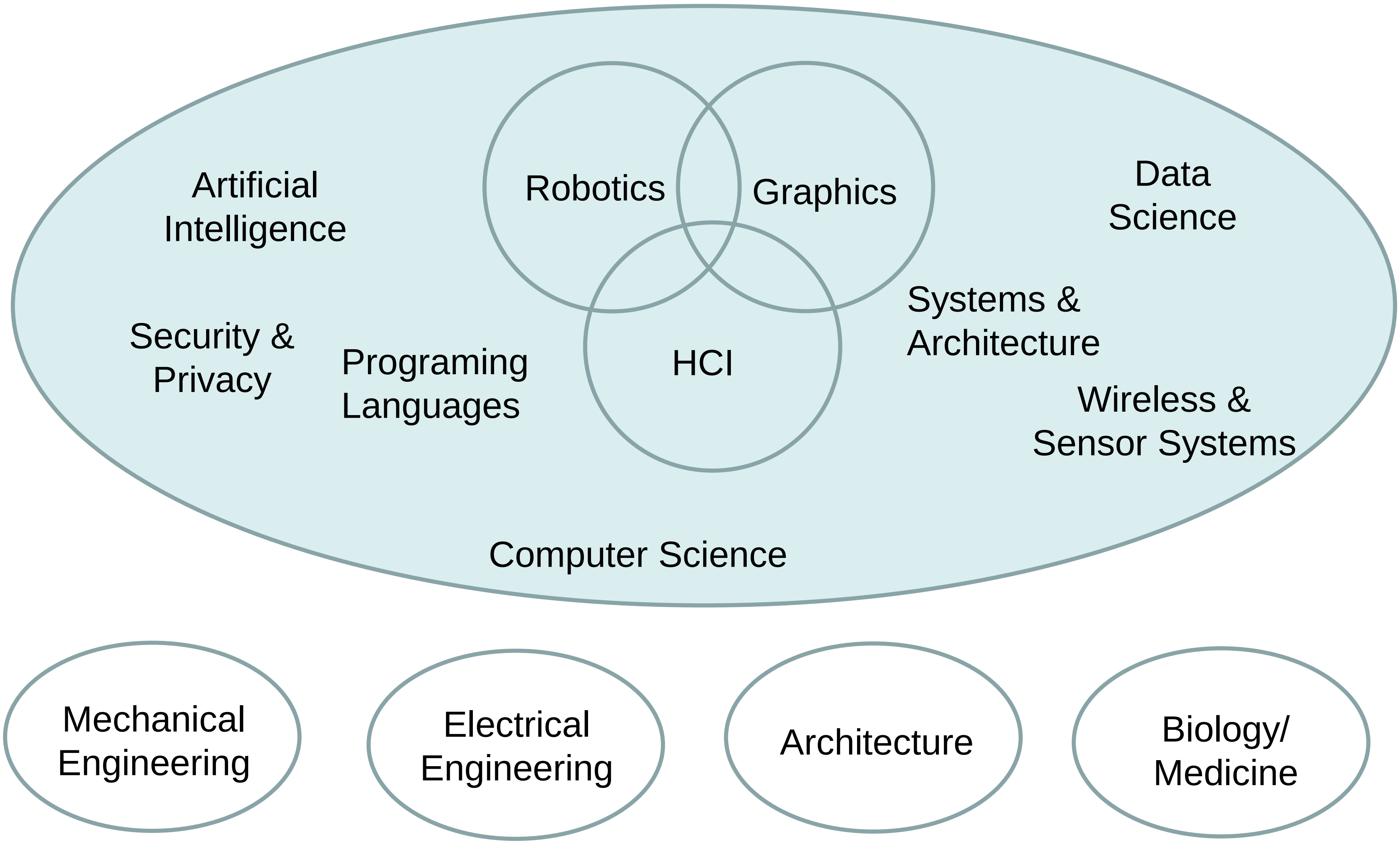




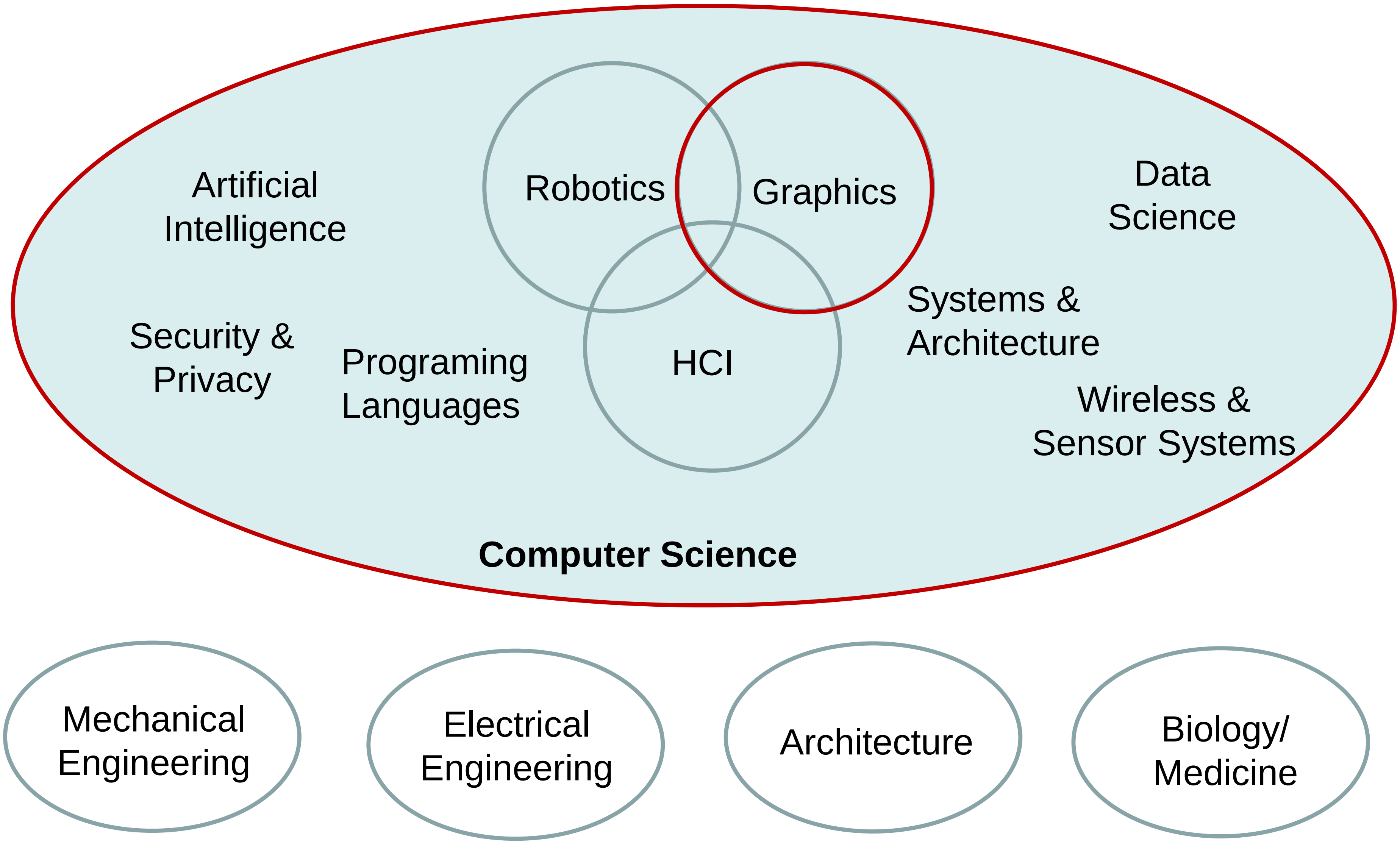
Geometry Processing

Performance-Driven Design

Active Research Areas

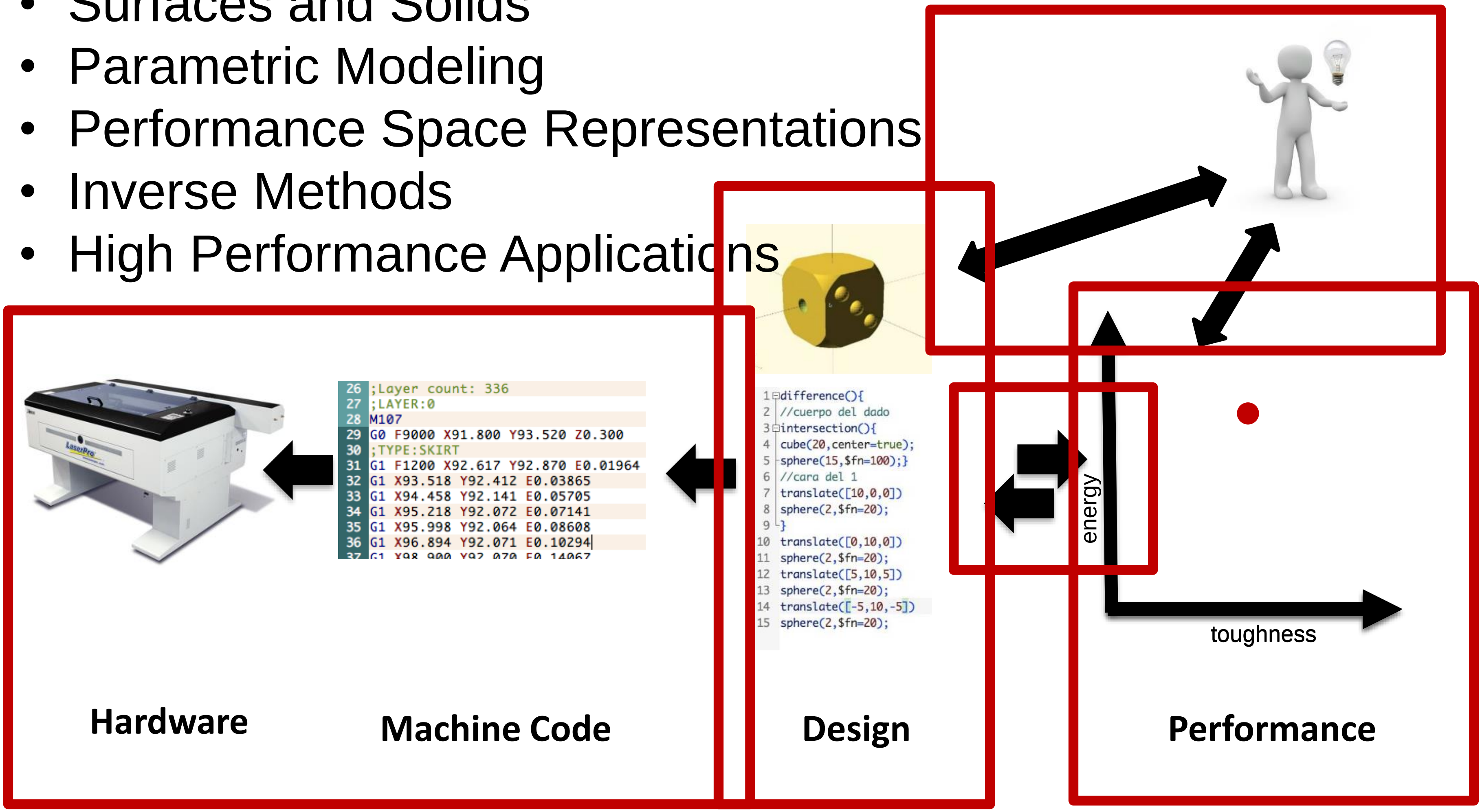


Our Perspective



What Will You Learn?

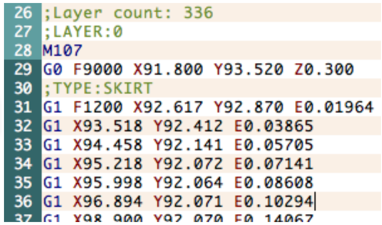
- Hardware and its abstractions
- Surfaces and Solids
- Parametric Modeling
- Performance Space Representations
- Inverse Methods
- High Performance Applications

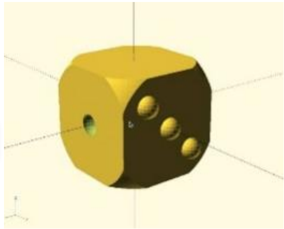


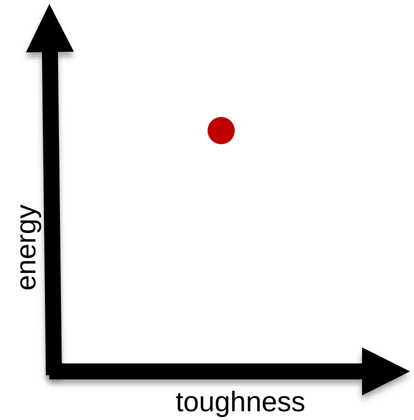
Course Schedule

9:00 am – 9:10 am	Introduction
9:10 am – 9:25 am	Hardware Review
9:25 am – 9:50 am	From Design to Machine Code
9:50 am – 10:20 am	Design Space Representations
10:20 am – 10:25 am	Performance Driven Design
10:25 am - 10:40 am	Break
10:40 am – 10:55 am	Performance Space Representation
10:55 am - 11:15 am	Inverse Methods
11:15 am - 11:35 am	Multi-objective Inverse Methods
11:35 am - 12:00 pm	Advanced Performance Driven Design
12:00 pm - 12:15 pm	Course Review

Computational Design Stack







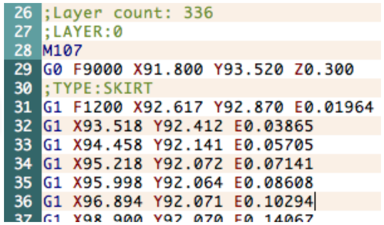
Hardware

Machine Code

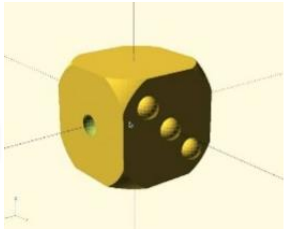
Design

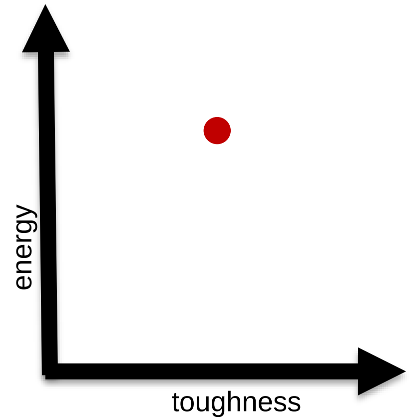
Performance

Computational Design Stack



```
26 ;Layer count: 336
27 ;LAYER:0
28 M107
29 G0 F9000 X91.800 Y93.520 Z0.300
30 ;TYPE:SKIRT
31 G1 F1200 X92.617 Y92.870 E0.01964
32 G1 X93.518 Y92.412 E0.03865
33 G1 X94.458 Y92.141 E0.05705
34 G1 X95.218 Y92.072 E0.07141
35 G1 X95.998 Y92.064 E0.08608
36 G1 X96.894 Y92.071 E0.10294
37 G1 X98.900 Y92.070 E0.14067
```





Hardware

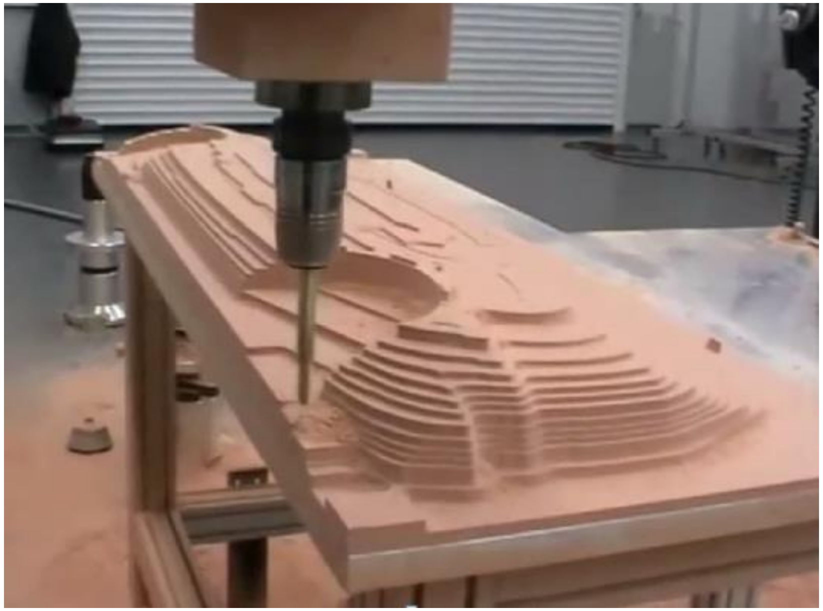
Machine Code

Design

Performance

Subtractive Manufacturing

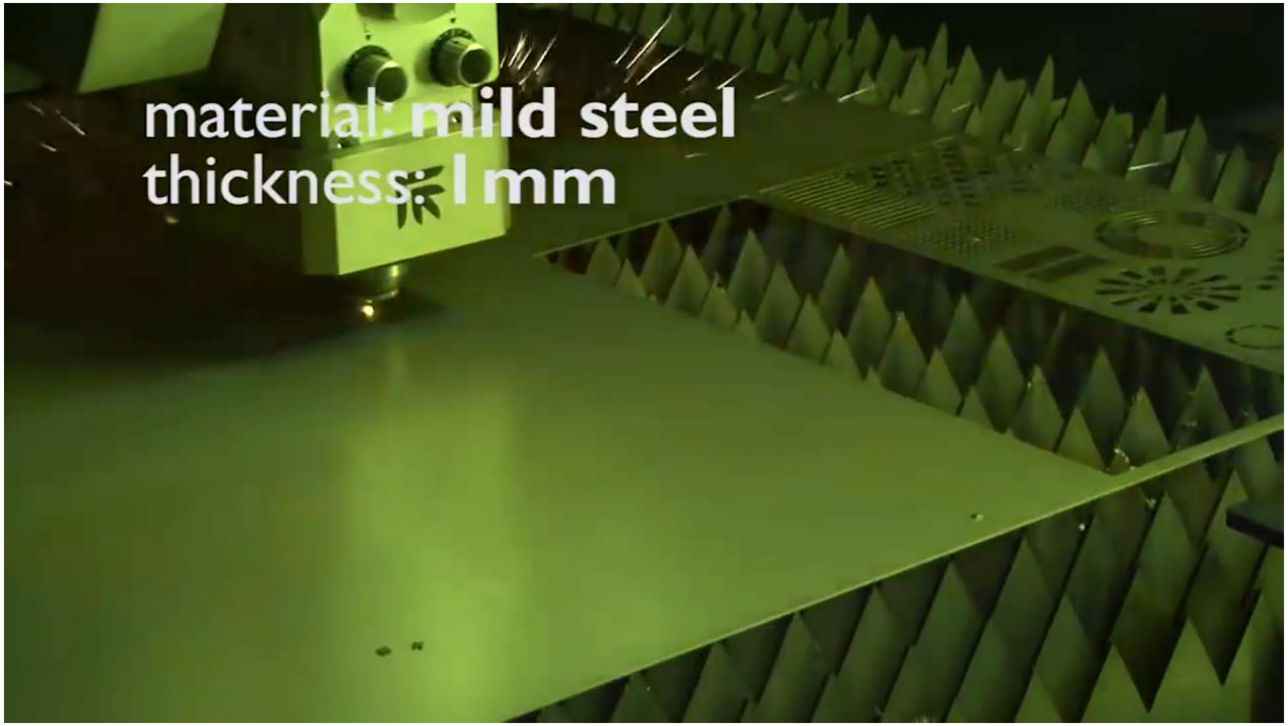
- Start with a block of material
- Remove material to obtain a given shape



Source: https://commons.wikimedia.org

Computer Numerical Control (CNC) Cutting

- Many different methods (e.g, laser cutting, water jetting)
- Input: 2D vector format
- Output: G-code



Source: https://en.wikipedia.org/wiki/Laser_cutting

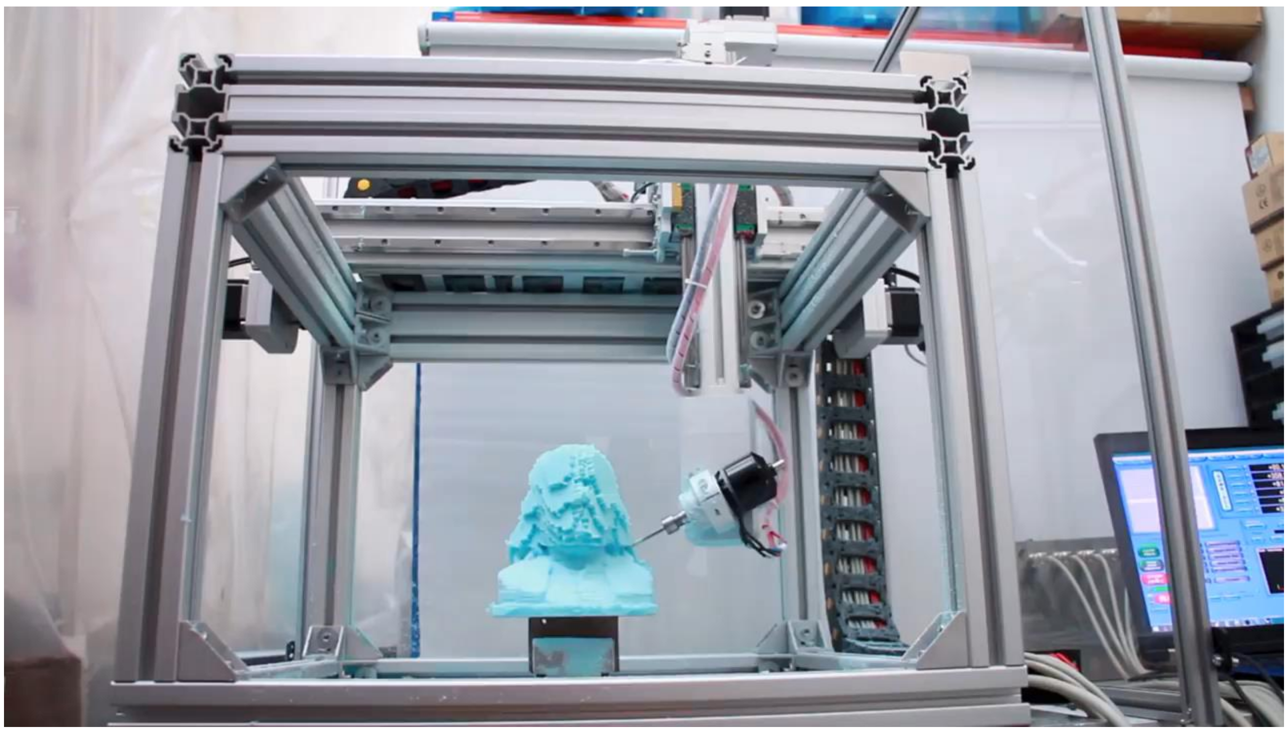
3-Axis CNC Milling/Engraving

- Input: 2.5D (heightfield)
- Output: G-code, multiple passes

Source: https://www.youtube.com/watch?v=uE-49w6JtTk

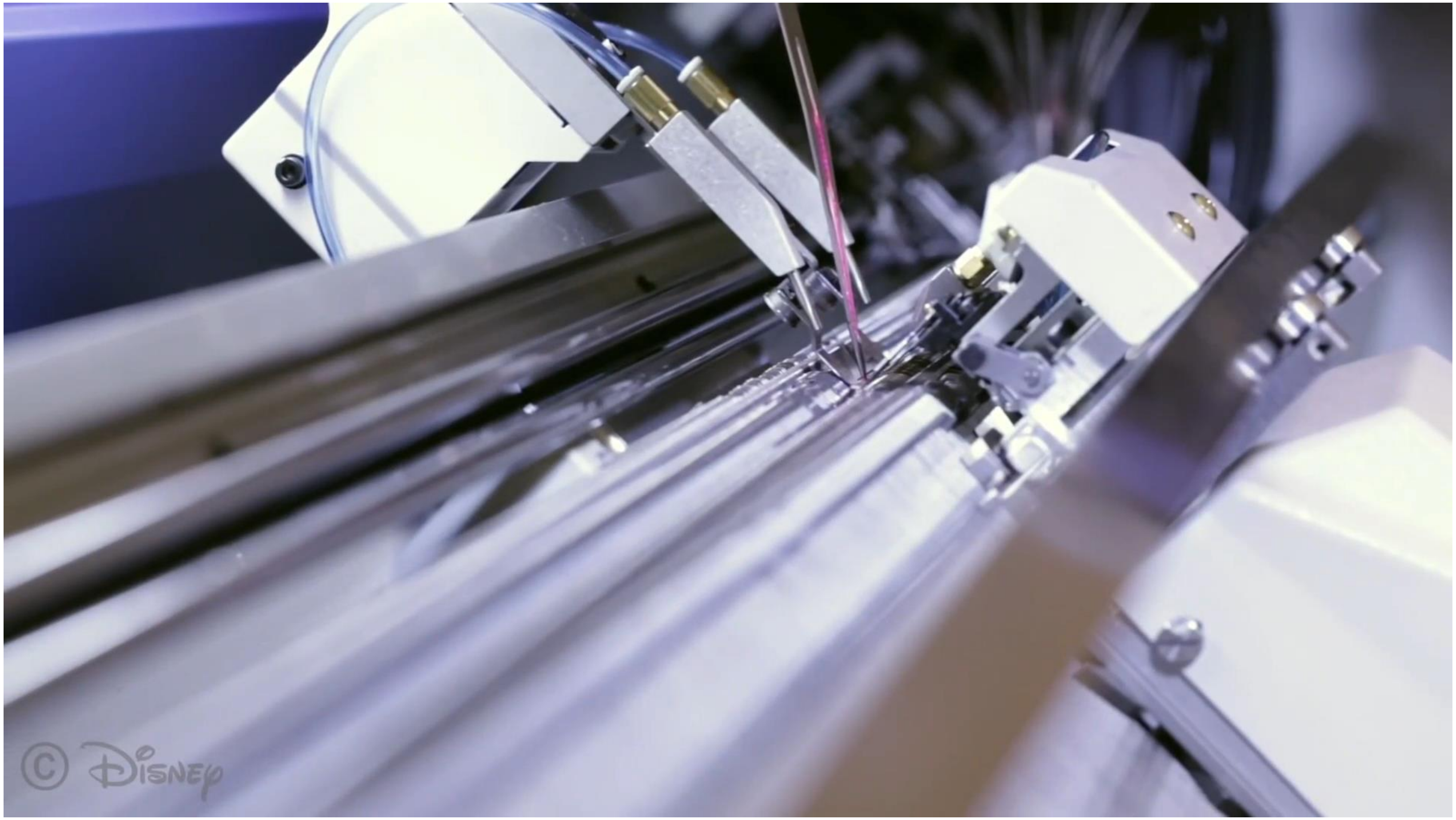
5-Axis CNC Milling

- Input: 3D Mesh
- Output: G-code, multiple passes



Source: https://www.youtube.com/watch?v=uE-49w6JtTk

Whole-garment Knitting



Source: ACM Siggraph

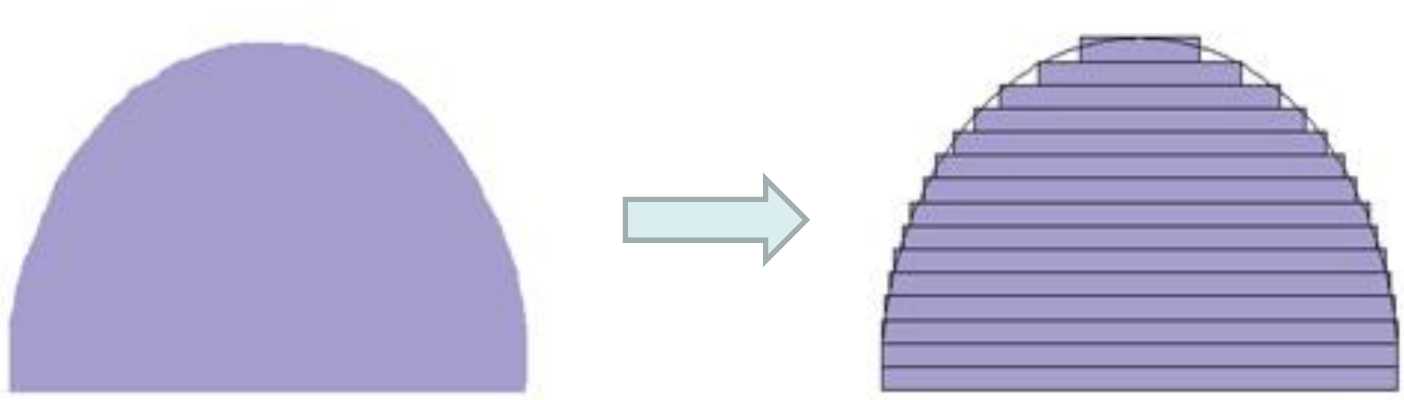
3D Printing Basics

3D Printing = Additive Manufacturing

Source: https://commons.wikimedia.org/wiki/File:3D_printing_on_replicator_2.webm

3D Printing Process

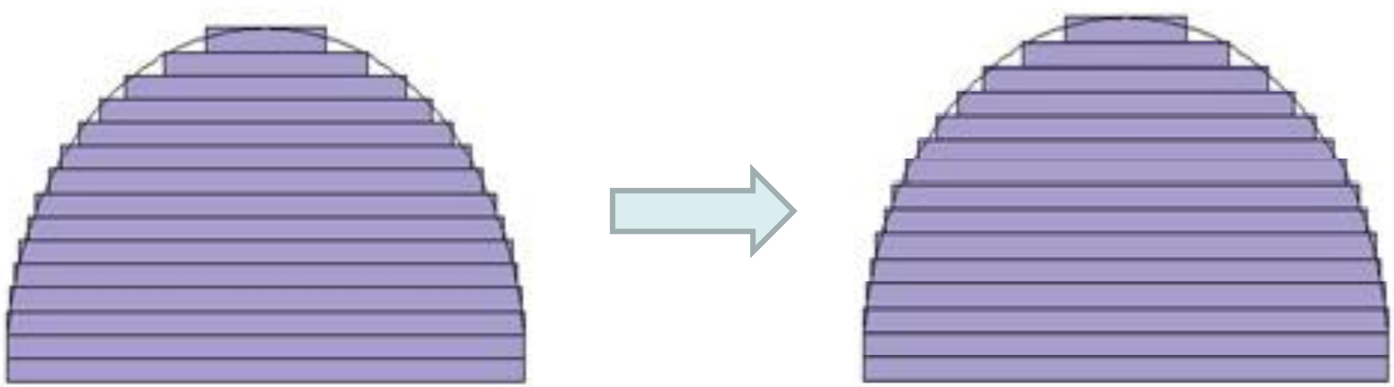
- Slice 3D model into layers



Source: https://commons.wikimedia.org

3D Printing Process

- Slice 3D model into layers
- Manufacture layers one by one (e.g., bottom-up)



Source: https://commons.wikimedia.org

Additive Manufacturing Processes

- Thermoplastic Extrusion
  - Fused deposition modeling (FDM)
- UV Curable Resins/thermosets
  - Stereolithography (SLA) & DLP Printing
  - Photopolymer Inkjet Printing
- Powders
  - Selective laser sintering (SLS)
  - Binder jetting/3D Printing
- Sheets
  - Laminated object manufacturing (LOM)

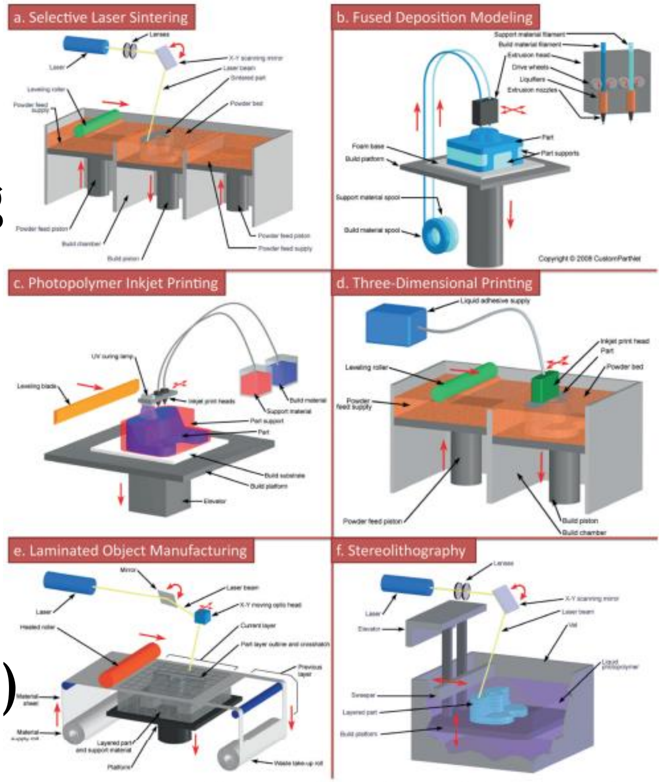


Figure 2. Various 3D printing techniques. a) Selective laser sintering (SLS). b) Fused deposition modeling (FDM, also termed "thermoplastic extrusion"). c) Photopolymer inkjet printing. d) Binder ketting, also trademarked as 3DP. e) Laminated object manufacturing (LOM). f) Stereolithography (SL). Images courtesy of CustomPartNet.com.

Additive Manufacturing Processes

- Thermoplastic Extrusion
  - Fused deposition modeling (FDM)
- UV Curable Resins/thermosets
  - Stereolithography (SLA) & DLP Printing
  - Photopolymer Inkjet Printing
- Powders
  - Selective laser sintering (SLS)
  - Binder jetting/3D Printing
- Sheets
  - Laminated object manufacturing (LOM)

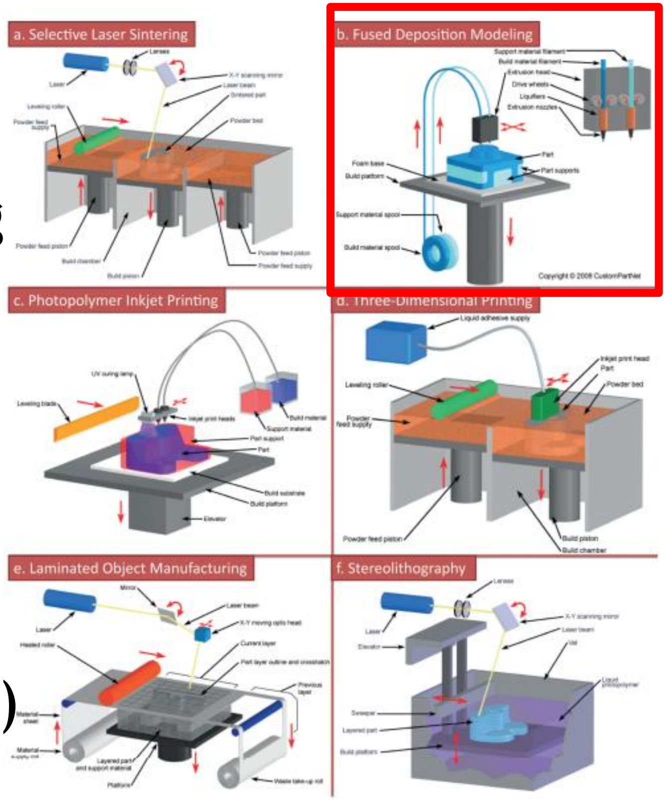
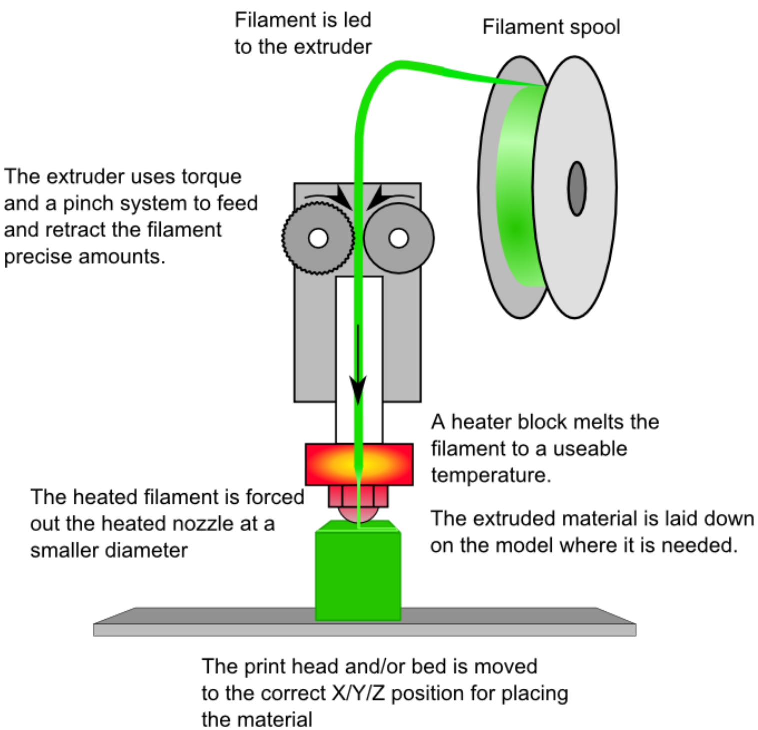


Figure 2. Various 3D printing techniques. a) Selective laser sintering (SLS). b) Fused deposition modeling (FDM, also termed "thermoplastic extrusion"). c) Photopolymer inkjet printing. d) Binder ketting, also trademarked as 3DP. e) Laminated object manufacturing (LOM). f) Stereolithography (SL). Images courtesy of CustomPartNet.com.

Fused Deposition Modeling (FDM)

- Fused Filament Fabrication (FFF)
- Filament is made of thermoplastic materials
  - e..g., ABS, polycarbonate, PLA
- Temporary support structure can be made from water-soluble material such as PVA
  - removed using heated sodium hydroxide solution



Source: http://reprap.org

Additive Manufacturing Processes

- Thermoplastic Extrusion
  - Fused deposition modeling (FDM)
- UV Curable Resins/thermosets
  - Stereolithography (SLA) & DLP Printing
  - Photopolymer Inkjet Printing
- Powders
  - Selective laser sintering (SLS)
  - Binder jetting/3D Printing
- Sheets
  - Laminated object manufacturing (LOM)

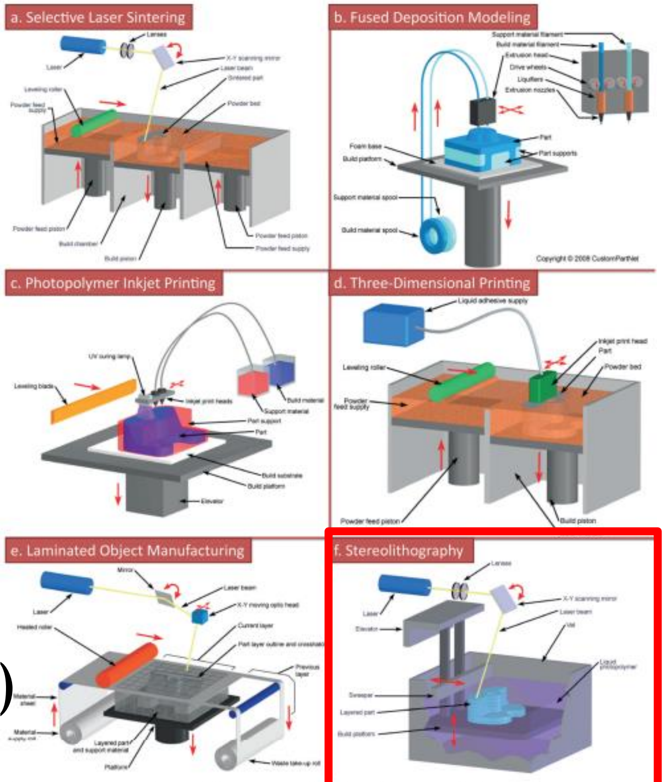
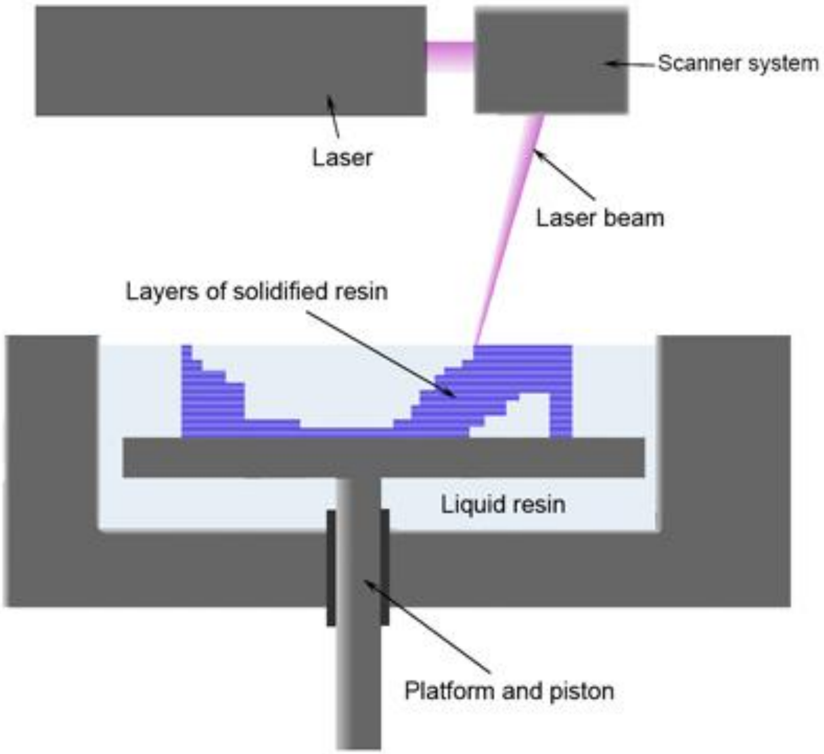


Figure 2. Various 3D printing techniques. a) Selective laser sintering (SLS). b) Fused deposition modeling (FDM, also termed "thermoplastic extrusion"). c) Photopolymer inkjet printing. d) Binder ketting, also trademarked as 3DP. e) Laminated object manufacturing (LOM). f) Stereolithography (SL). Images courtesy of CustomPartNet.com.

Stereolithography (SLA)

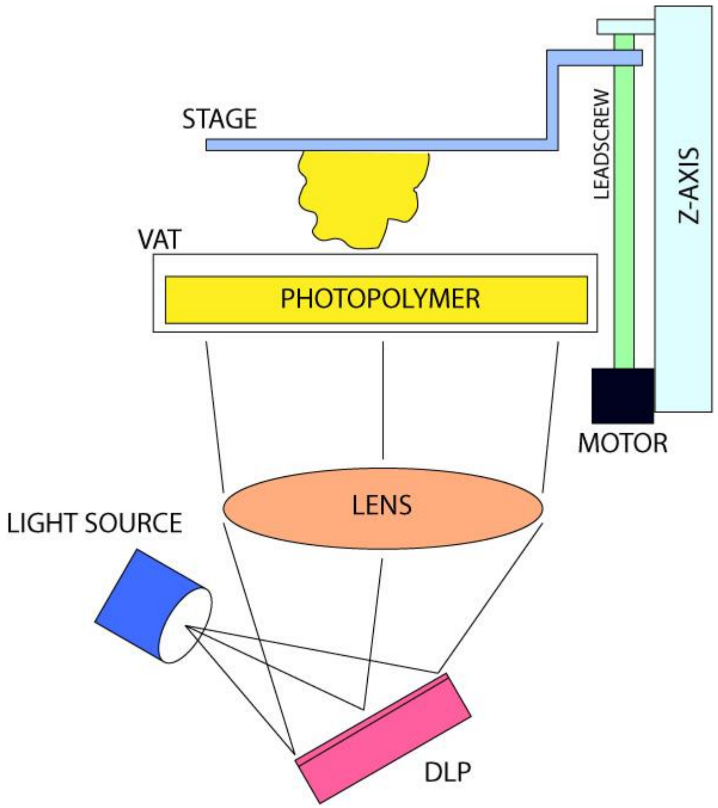
- SLA uses liquid ultraviolet curable photopolymer resin
- Laser beam traces one layer on the surface of the resin
- Laser light cures and solidifies the layer



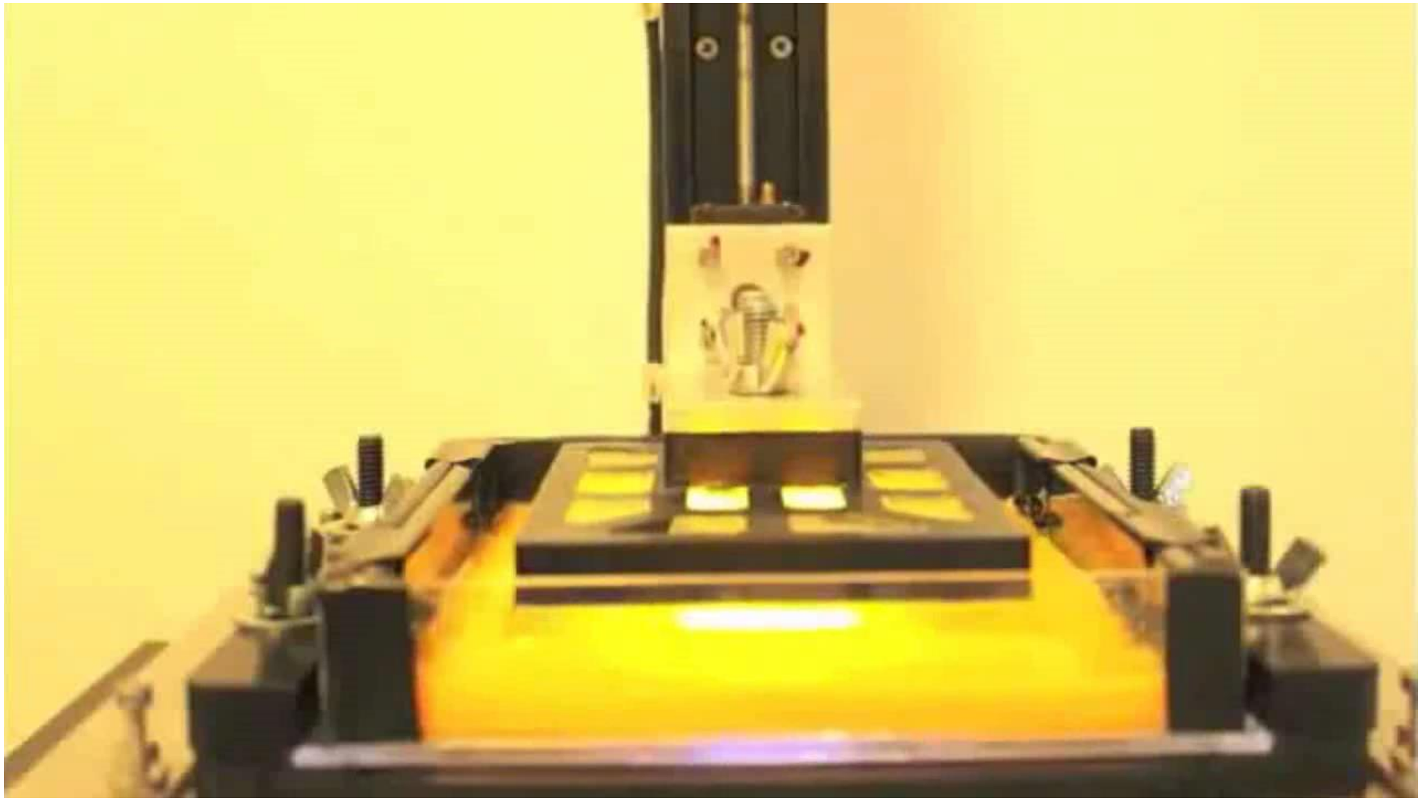
Source: http://en.wikipedia.org/

Digital Light Projector (DLP) 3D Printing

- DLP 3D printer uses liquid ultraviolet curable photopolymer resin
- DLP exposes and solidifies one layer at a time on the surface of the resin



DLP 3D Printing Process



Additive Manufacturing Processes

- Thermoplastic Extrusion
  - Fused deposition modeling (FDM)
- UV Curable Resins/thermosets
  - Stereolithography (SLA) & DLP Printing
  - Photopolymer Inkjet Printing
- Powders
  - Selective laser sintering (SLS)
  - Binder jetting/3D Printing
- Sheets
  - Laminated object manufacturing (LOM)

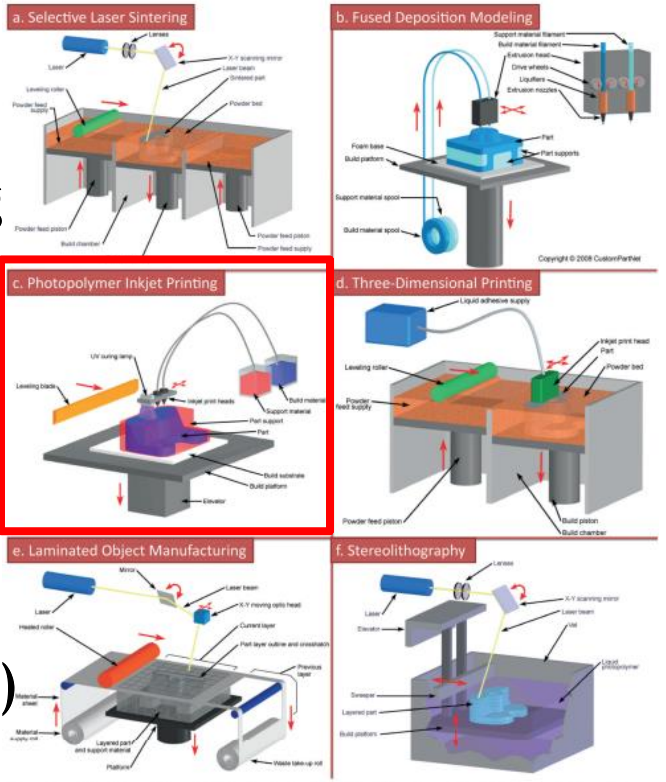
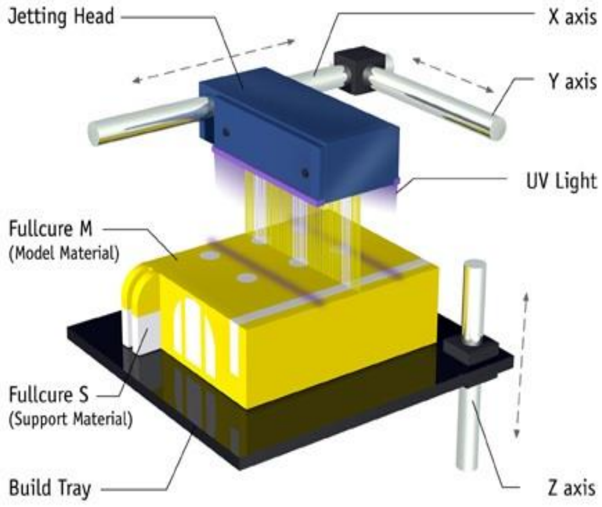


Figure 2. Various 3D printing techniques. a) Selective laser sintering (SLS). b) Fused deposition modeling (FDM, also termed "thermoplastic extrusion"). c) Photopolymer inkjet printing. d) Binder ketting, also trademarked as 3DP. e) Laminated object manufacturing (LOM). f) Stereolithography (SL). Images courtesy of CustomPartNet.com.

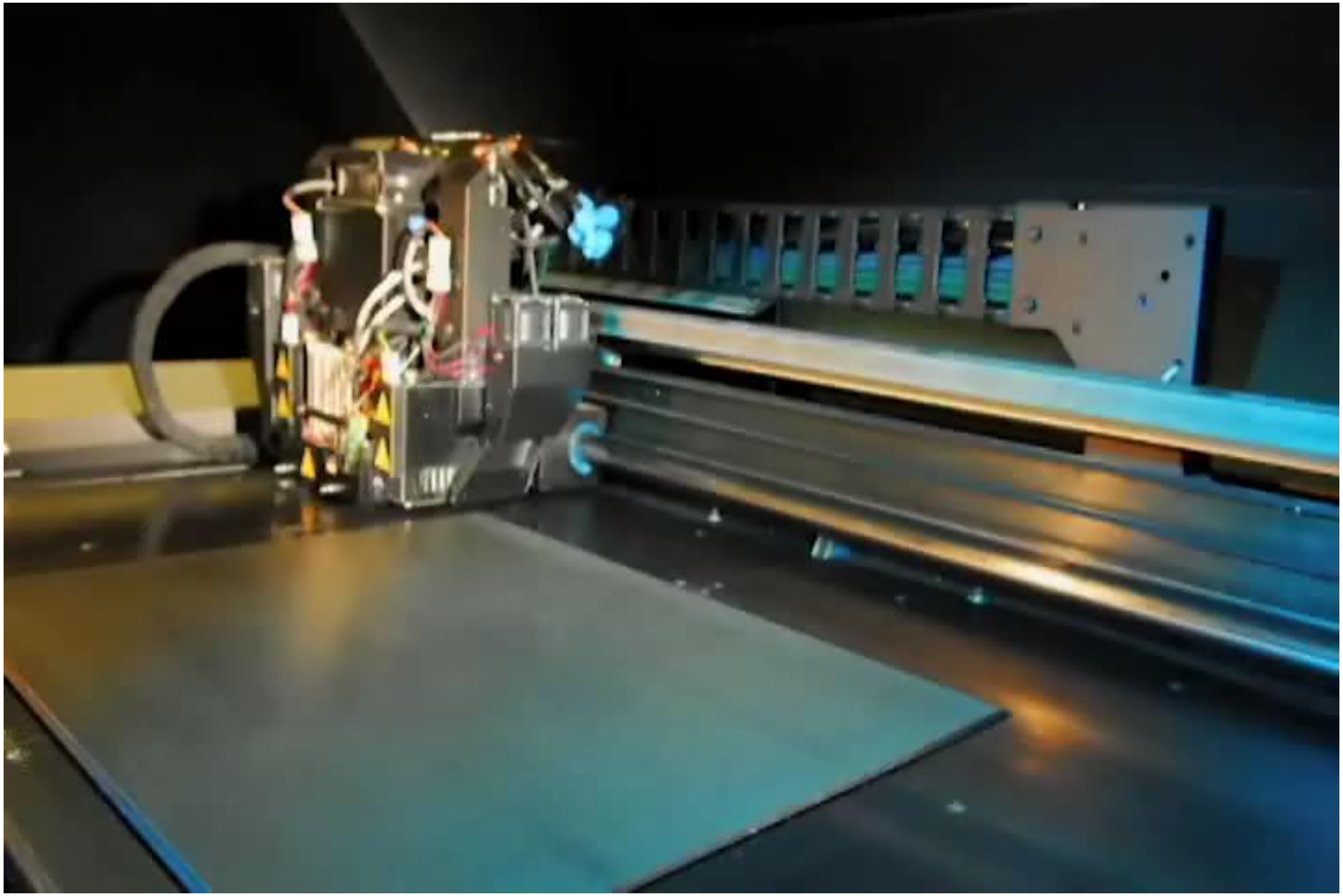
Photopolymer Inkjet Printing

- Inkjet printhead jets liquid photopolymer and support material
- UV light cures photopolymer and support material
- Excess material is removed using a roller
- The platform descends by one layer



Source: http://www.engatech.com/Objet-3D-Printing-Technology.asp

Printing Process



Source: https://www.youtube.com/watch?v=XLLq9SwSTpM

Sample Fabricated Objects

Source: Stratasys

Additive Manufacturing Processes

- Thermoplastic Extrusion
  - Fused deposition modeling (FDM)
- UV Curable Resins/thermosets
  - Stereolithography (SLA) & DLP Printing
  - Photopolymer Inkjet Printing
- Powders
  - Selective laser sintering (SLS)
  - Binder jetting/3D Printing
- Sheets
  - Laminated object manufacturing (LOM)

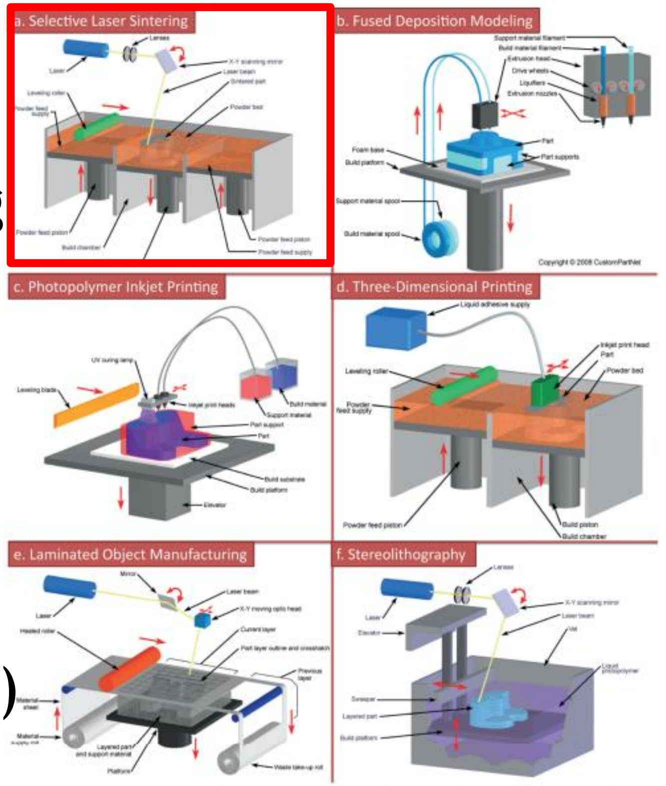
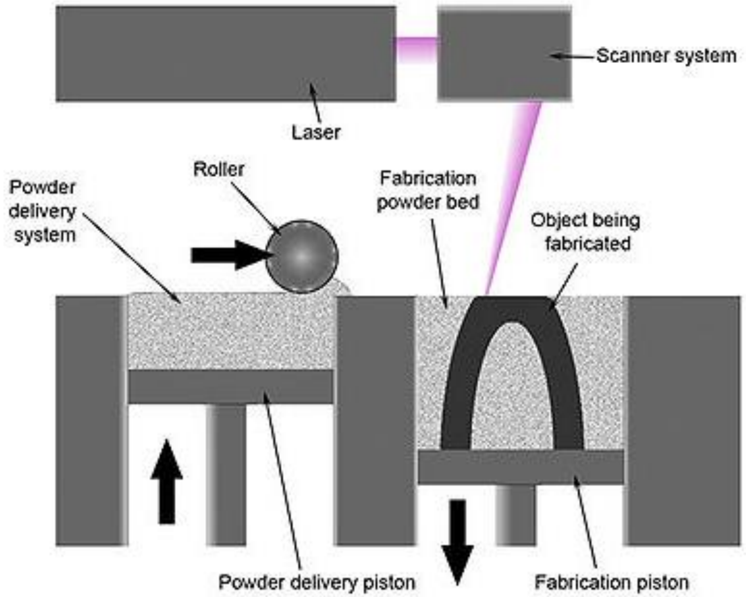


Figure 2. Various 3D printing techniques. a) Selective laser sintering (SLS). b) Fused deposition modeling (FDM, also termed "thermoplastic extrusion"). c) Photopolymer inkjet printing. d) Binder ketting, also trademarked as 3DP. e) Laminated object manufacturing (LOM). f) Stereolithography (SL). Images courtesy of CustomPartNet.com.

Selective Laser Sintering (SLS)
Direct Metal Laser Sintering (DMLS)

- SLS and DMLS use a bed of small particles (made of plastic, metal, ceramic, or glass)
- High-power laser traces one layer on the surface of the powder bed melting/fusing the particles

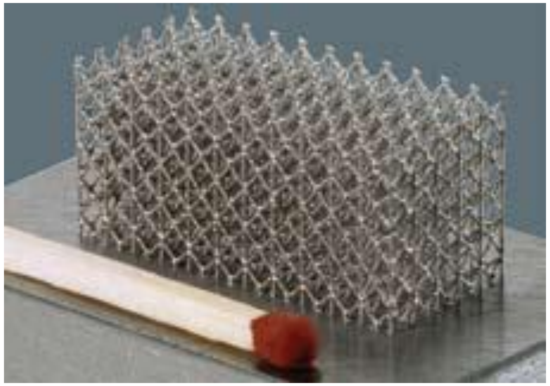


Source: http://en.wikipedia.org/wiki/Selective_laser_sintering

SLS & DMLS Process

Source: https://www.youtube.com/watch?v=BZLGLzyMKn4

Sample Fabricated Parts



Source: http://www.bridgesmathart.org , http://www.freedomofcreation.com

Additive Manufacturing Processes

- Thermoplastic Extrusion
  - Fused deposition modeling (FDM)
- UV Curable Resins/thermosets
  - Stereolithography (SLA) & DLP Printing
  - Photopolymer Inkjet Printing
- Powders
  - Selective laser sintering (SLS)
  - Binder jetting/3D Printing
- Sheets
  - Laminated object manufacturing (LOM

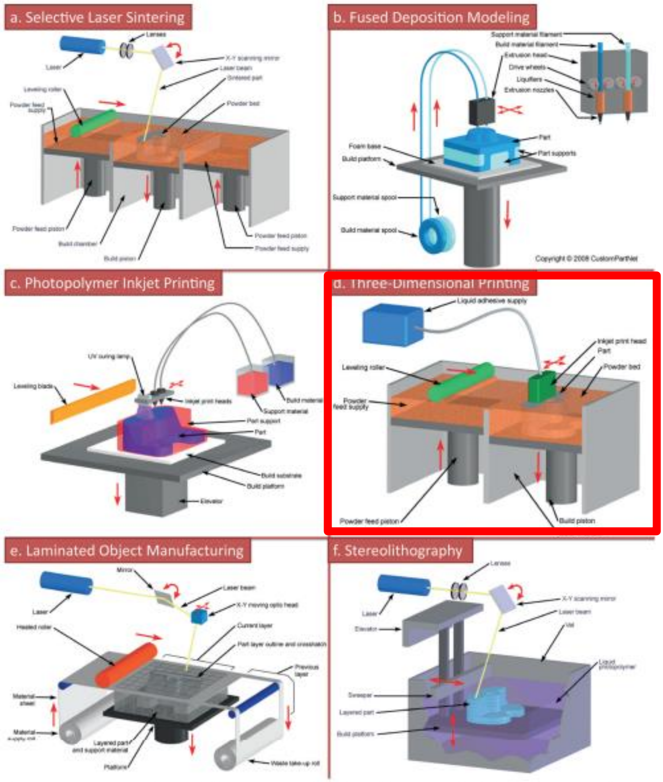
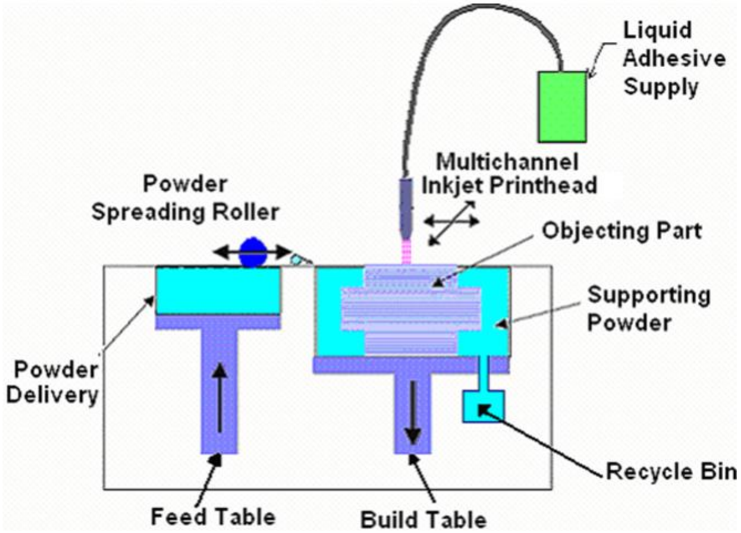


Figure 2. Various 3D printing techniques. a) Selective laser sintering (SLS). b) Fused deposition modeling (FDM, also termed "thermoplastic extrusion"). c) Photopolymer inkjet printing. d) Binder ketting, also trademarked as 3DP. e) Laminated object manufacturing (LOM). f) Stereolithography (SL). Images courtesy of CustomPartNet.com.

Binder Jet 3D Printing

- This method uses a bed of small plaster particles
- Inkjet printhead prints with liquid (possibly colored) adhesive one layer on the surface of the powder bed fusing the particles
- The platform descends by one layer and more material is added

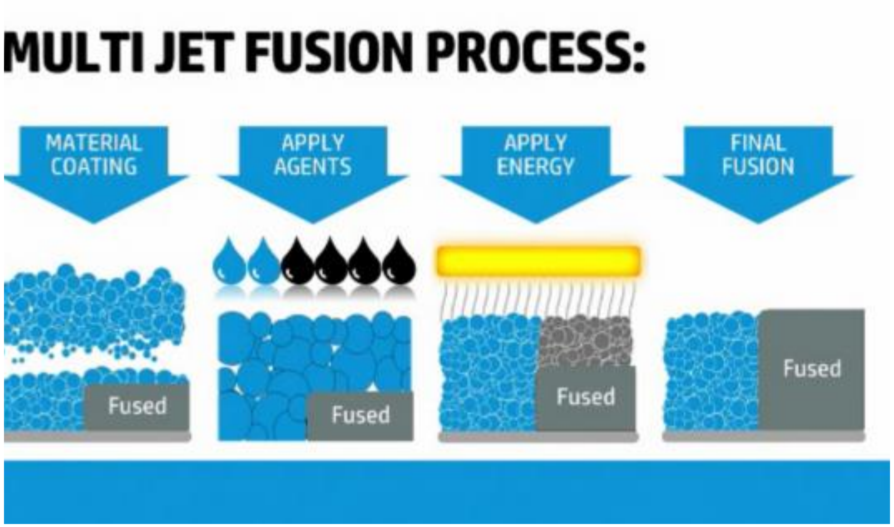


Source: Zhou and Lu, 2011

Binder Jet 3D Printing Process

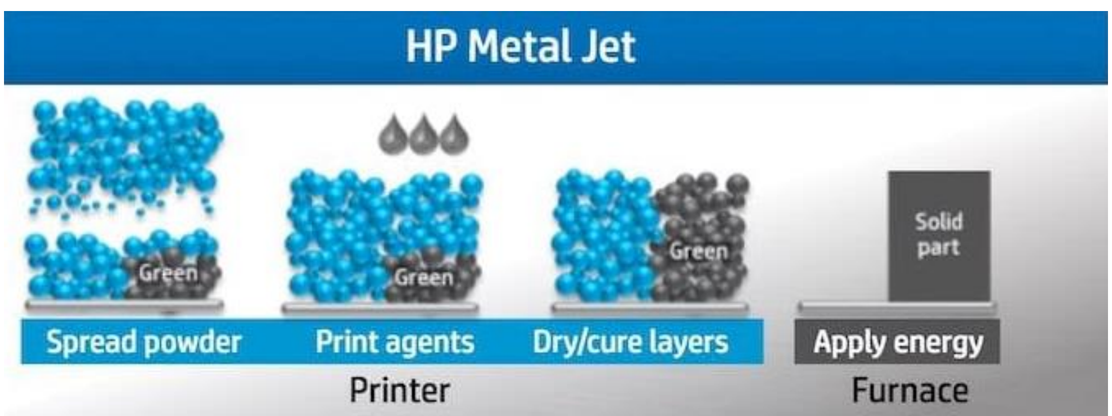
Source: https://www.youtube.com/watch?v=GnFxujCyD70

Binder Jet: Plastics (HP)



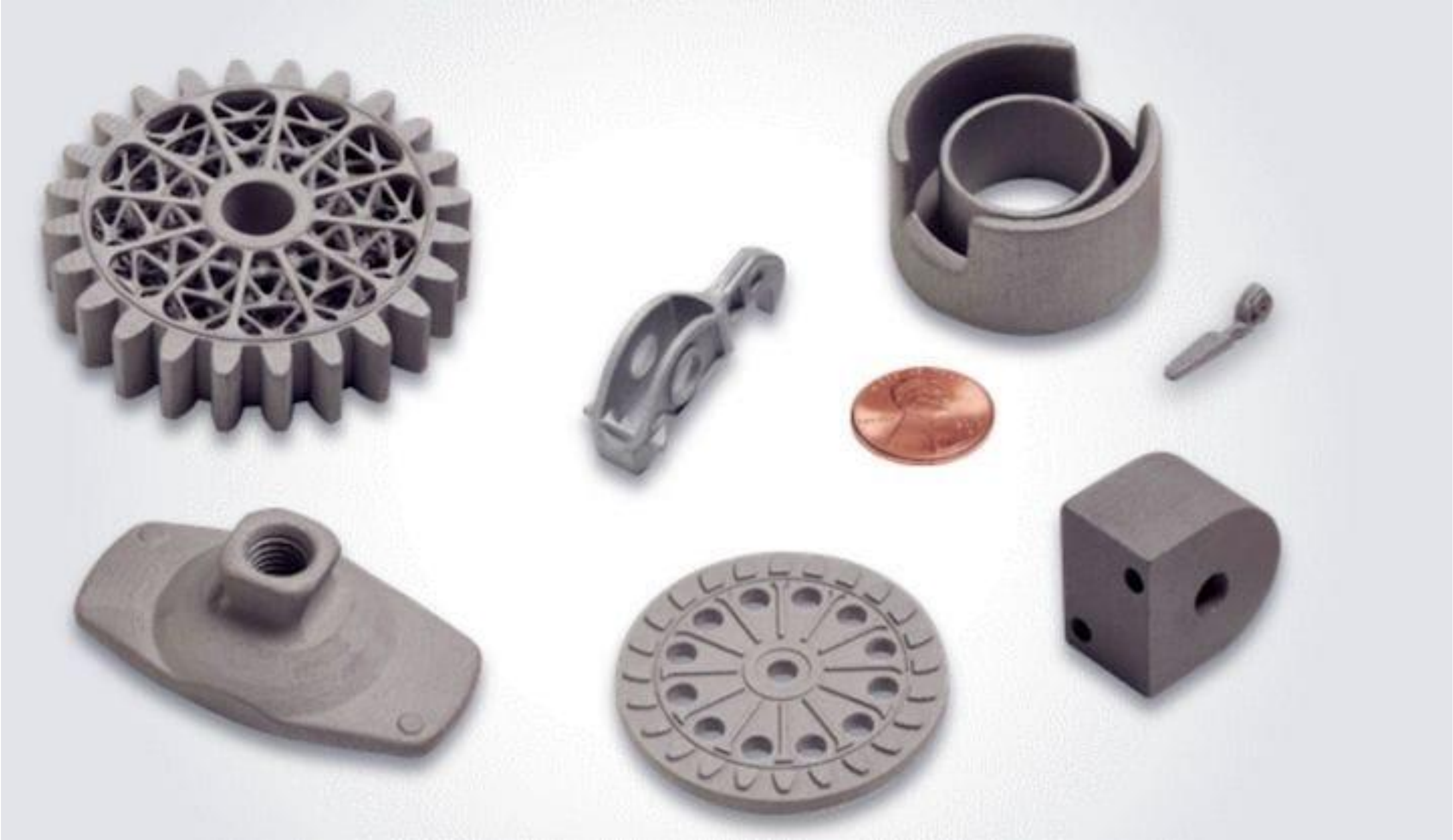
Source: HP

Binder Jet: Metals (HP)



Source: HP

Binder Jet: Metals (HP)



Source: HP

Additive Manufacturing Processes

- Thermoplastic Extrusion
  - Fused deposition modeling (FDM)
- UV Curable Resins/thermosets
  - Stereolithography (SLA) & DLP Printing
  - Photopolymer Inkjet Printing
- Powders
  - Selective laser sintering (SLS)
  - Binder jetting/3D Printing
- Sheets
  - Laminated object manufacturing (LOM)

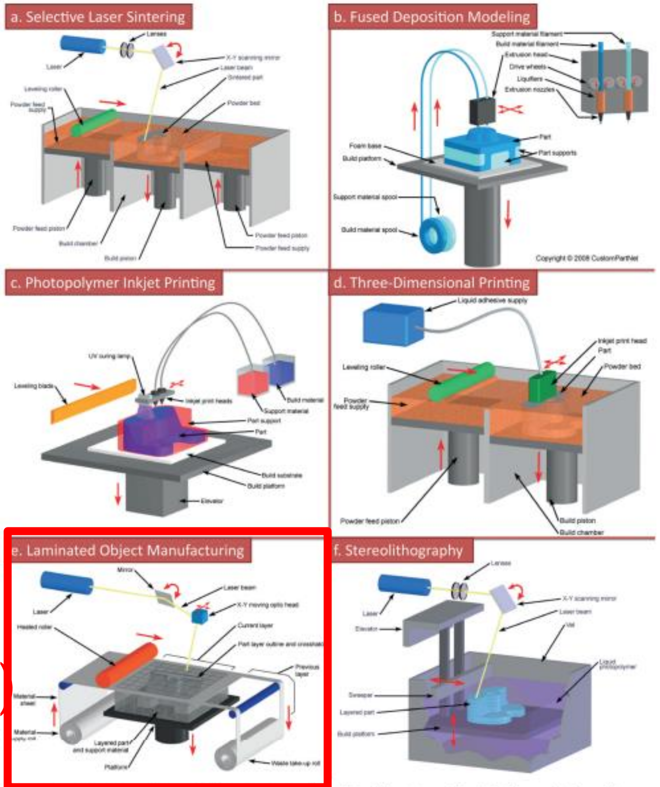
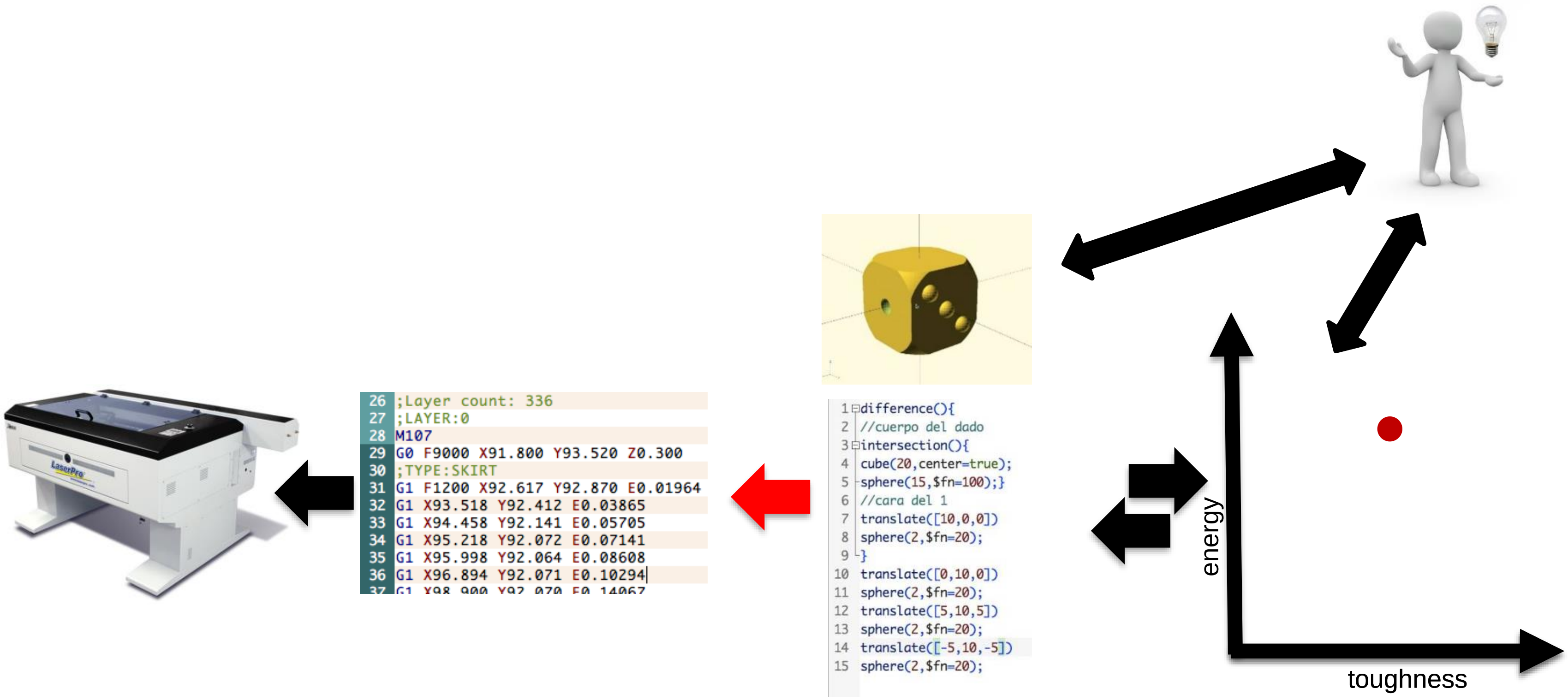


Figure 2. Various 3D printing techniques. a) Selective laser sintering (SLS). b) Fused deposition modeling (FDM, also termed "thermoplastic extrusion"). c) Photopolymer inkjet printing. d) Binder ketting, also trademarked as 3DP. e) Laminated object manufacturing (LOM). f) Stereolithography (SL). Images courtesy of CustomPartNet.com.

Course Schedule

9:00 am – 9:10 am	Introduction
9:10 am – 9:25 am	Hardware Review
9:25 am – 9:50 am	From Design to Machine Code
9:50 am – 10:20 am	Design Space Representations
10:20 am – 10:25 am	Performance Driven Design
10:25 am - 10:40 am	Break
10:40 am – 10:55 am	Performance Space Representation
10:55 am - 11:15 am	Inverse Methods
11:15 am - 11:35 am	Multi-objective Inverse Methods
11:35 am - 12:00 pm	Advanced Performance Driven Design
12:00 pm - 12:15 pm	Course Review

Computational Design and Fabrication Pipeline



Hardware **Machine Code** **Design** **Performance**

3D Printing Software Pipeline

Input Model

↓

Orientation and Positioning

↓

Support Structures

↓

Slicing

↓

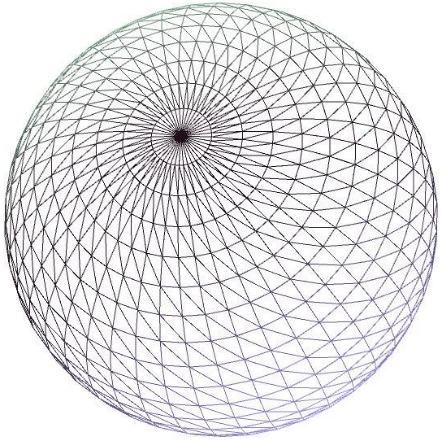
Path Planning

↓

Machine Instructions

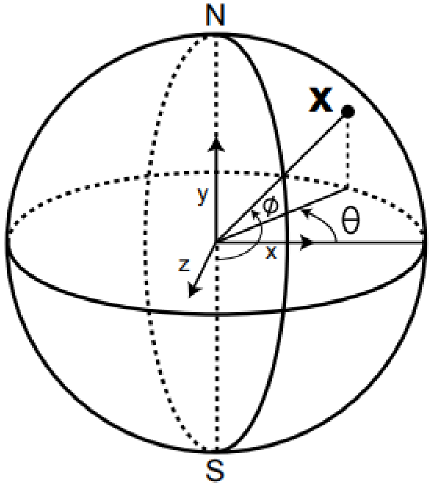
Surface Representations

Polygonal Mesh



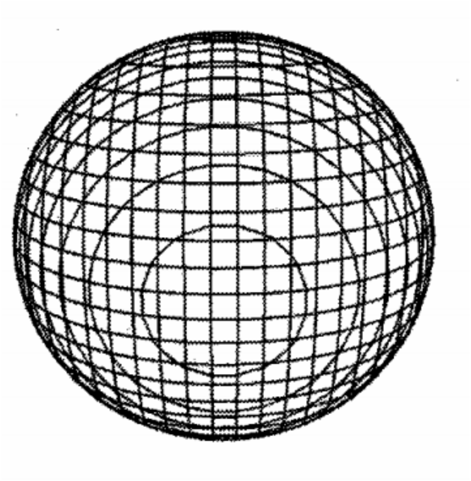
e.g. triangle soup

Parametric Surface



$$x = r \sin\phi \cos\theta,$$
$$y = -r \cos\phi,$$
$$z = -r \sin\phi \sin\theta,$$

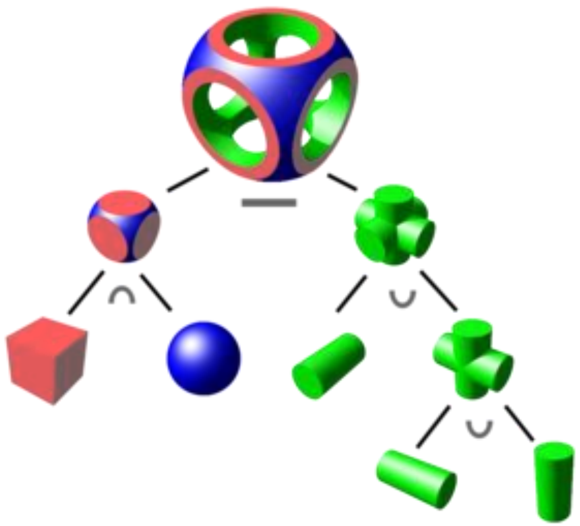
Implicit Surface



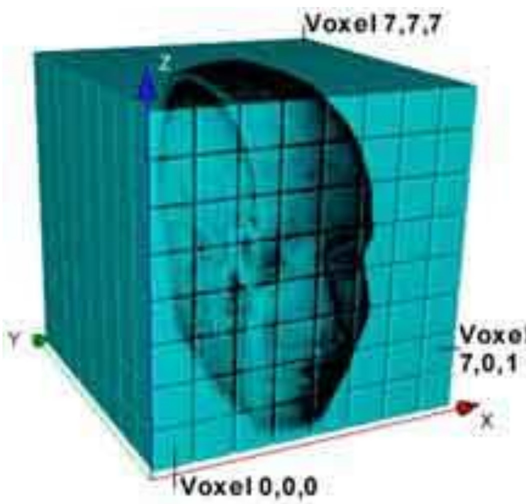
$$x^2 + y^2 + z^2 - r^2 = 0.$$

Volumetric Representations

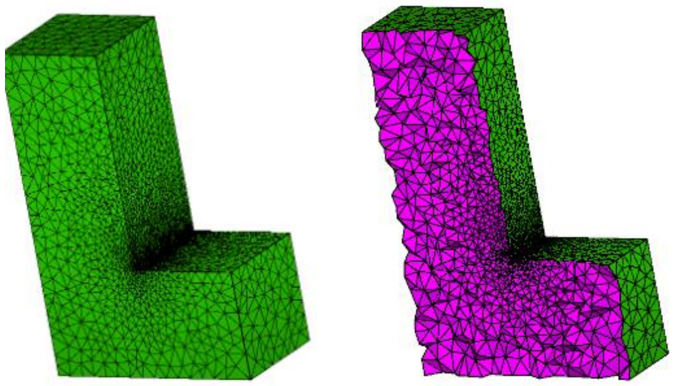
Constructive Solid Geometry



Voxel Grid

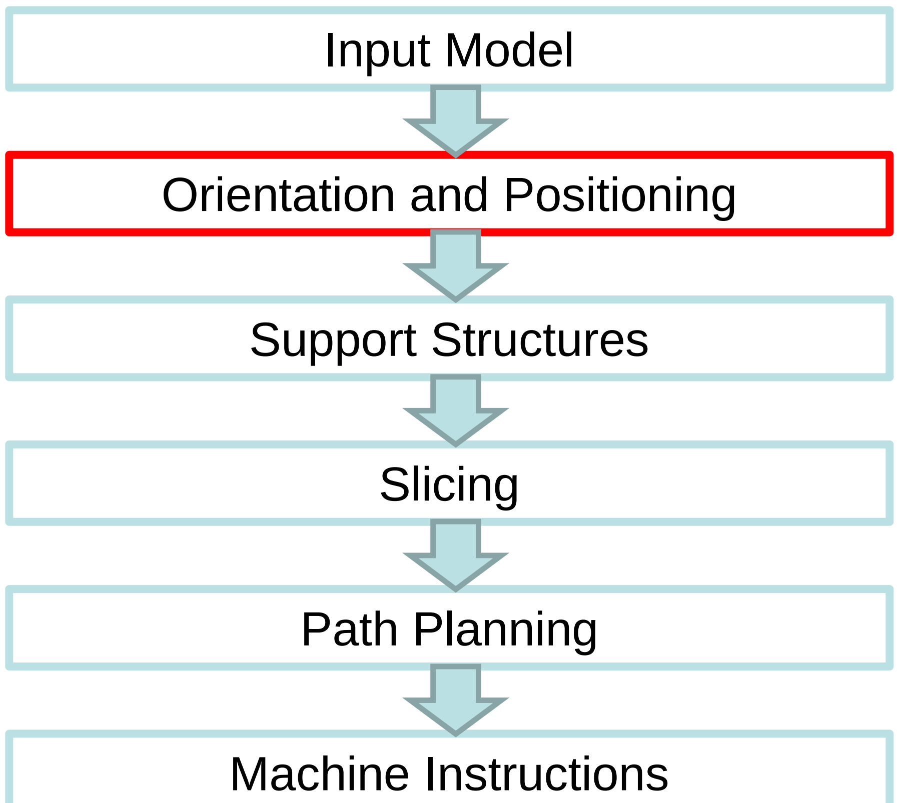


Tetrahedral Mesh

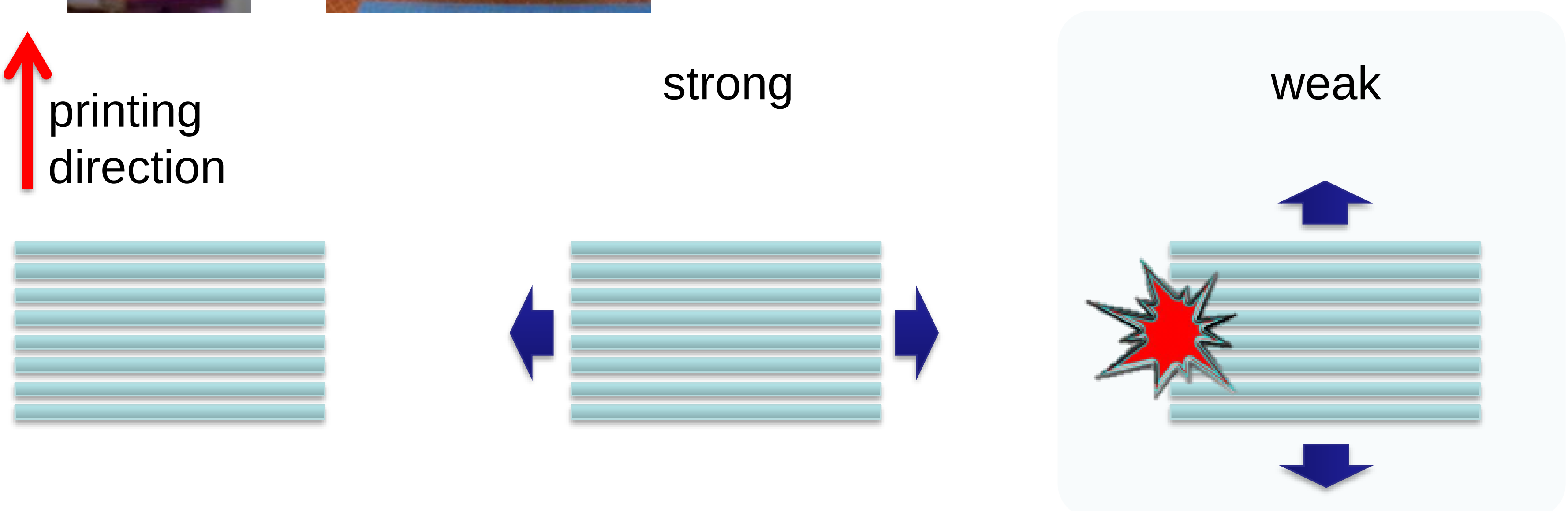


Source: Wikipedia, Tetgen.org, www.volumegraphics.com

3D Printing Software Pipeline

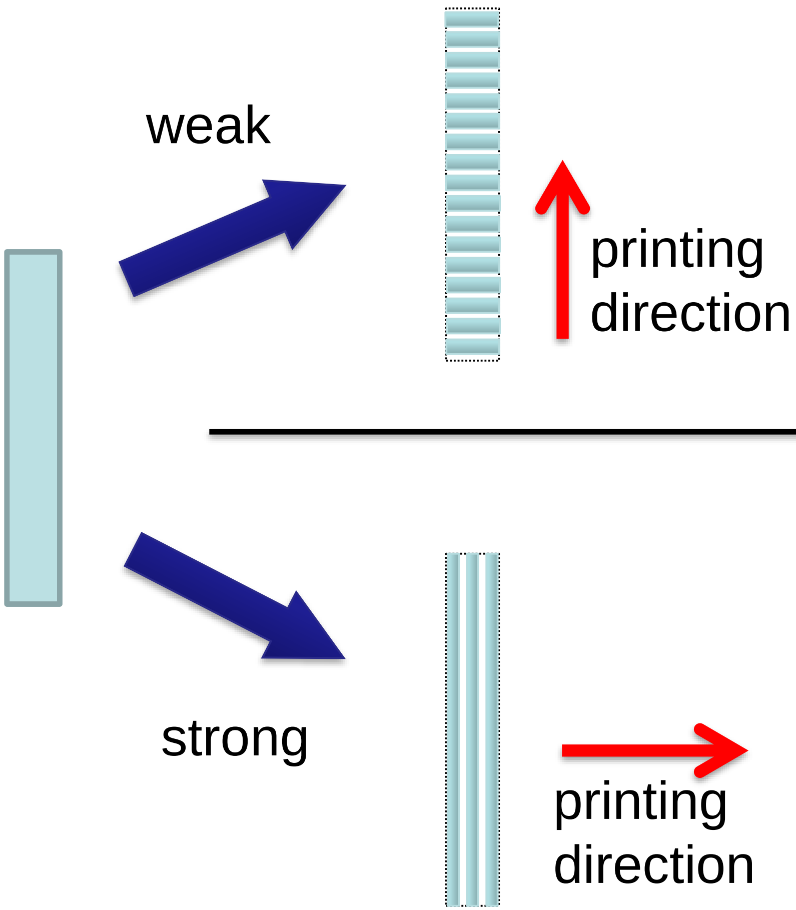


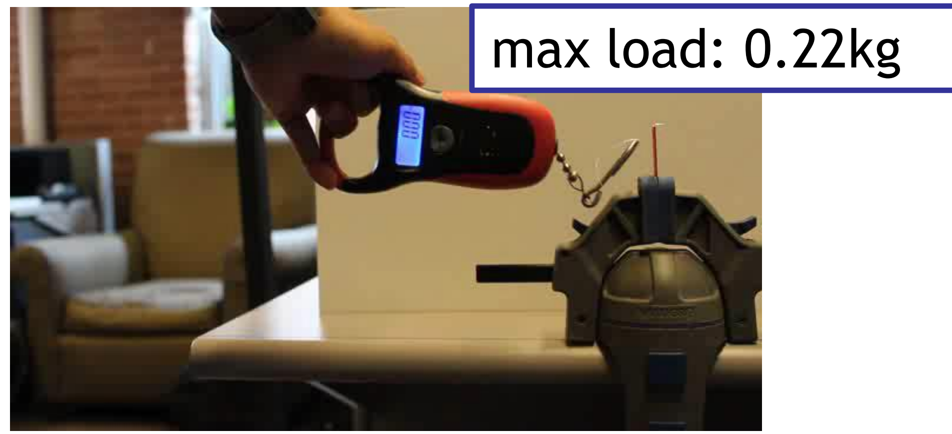
Model Orientation: Mechanical Properties

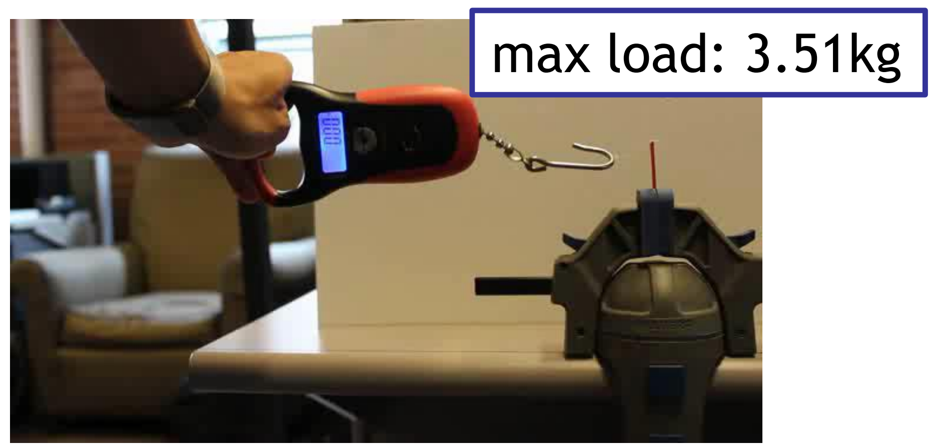


Source: ACM Siggraph

Model Orientation: Mechanical Properties



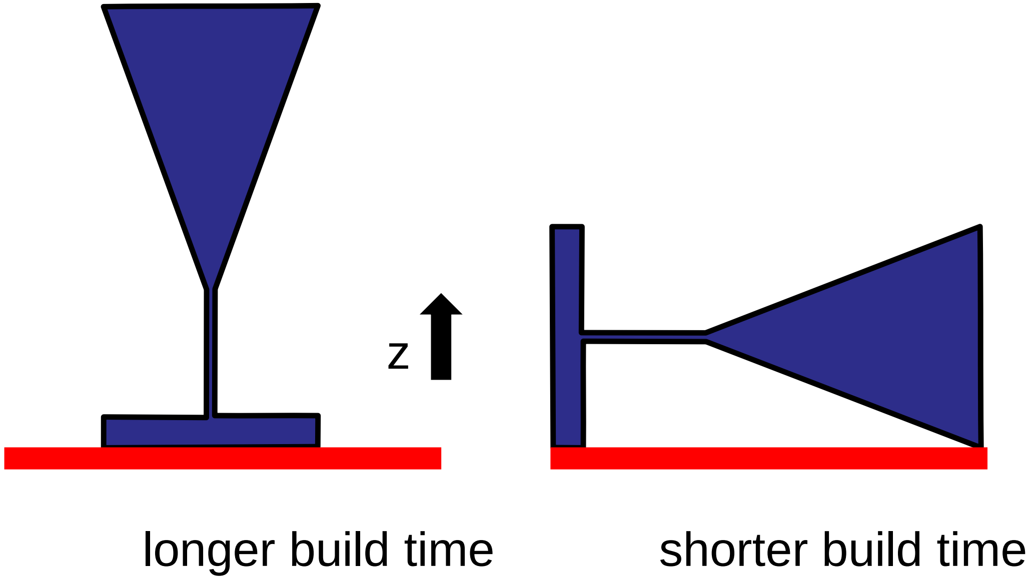




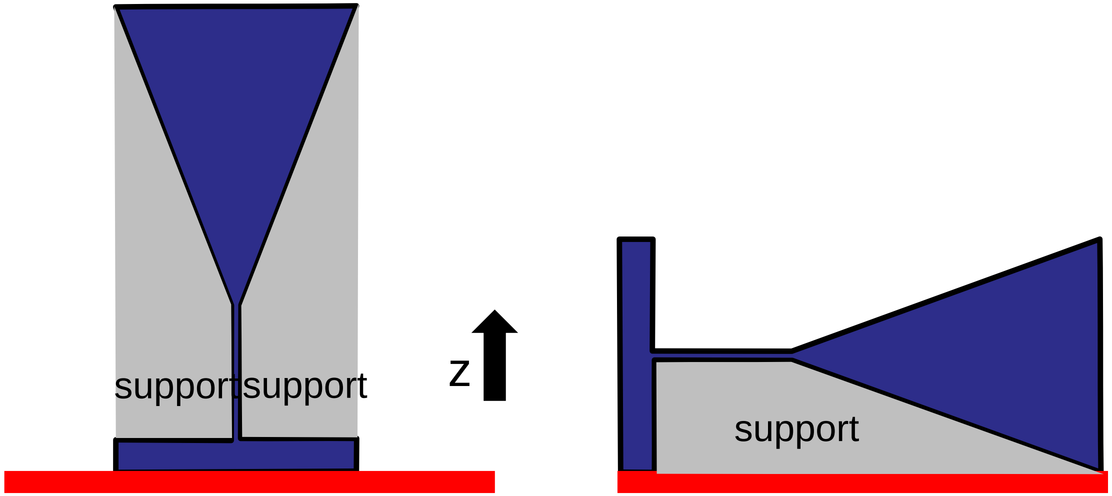
Source: ACM Siggraph

Model Orientation: Build Time

- Build speed is slower for the verical direction compared to the horizontal direction



Model Orientation: Support Volume

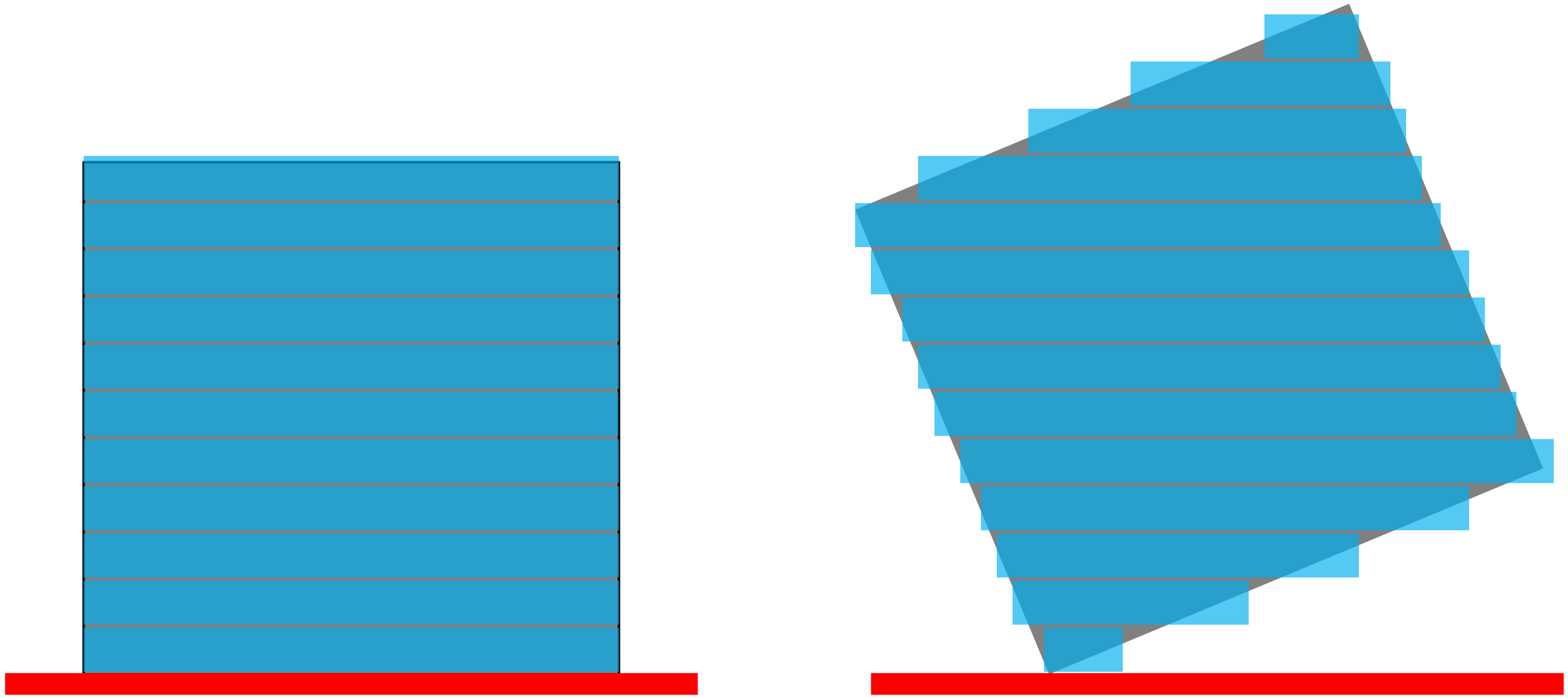


more support volume

less support volume

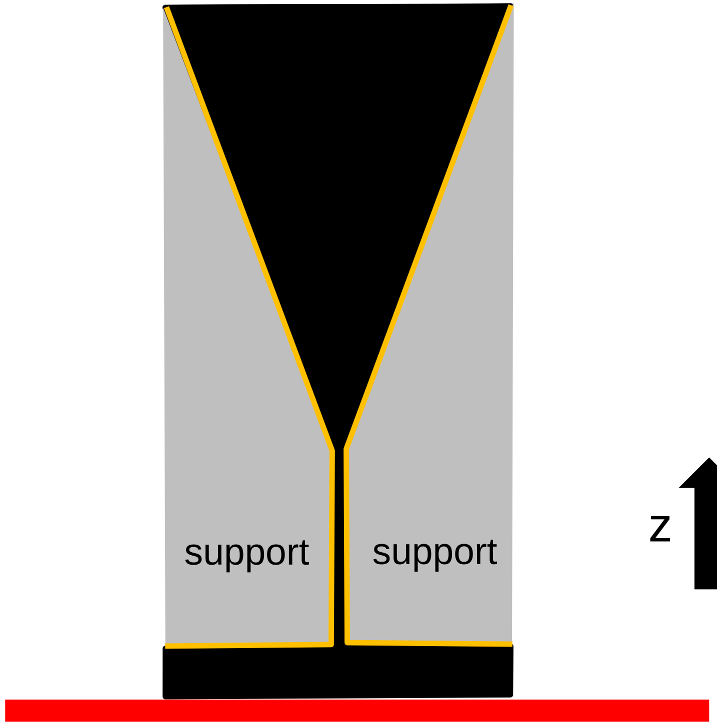
Model Orientation: Surface Accuracy

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

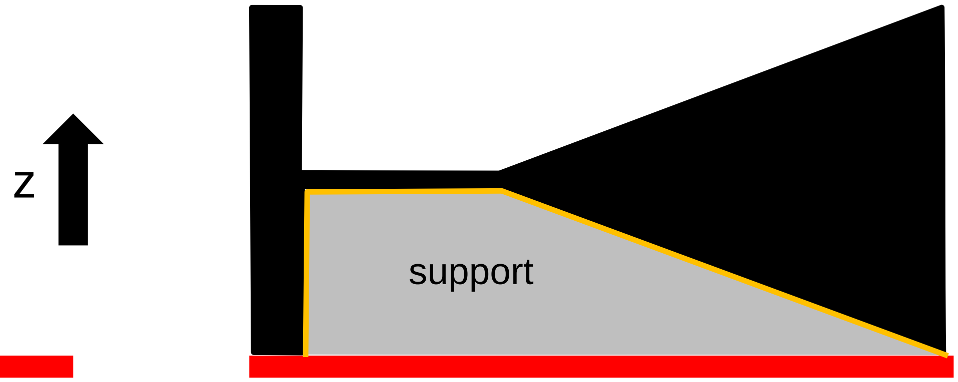


Model Orientation: Support Contact Area

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



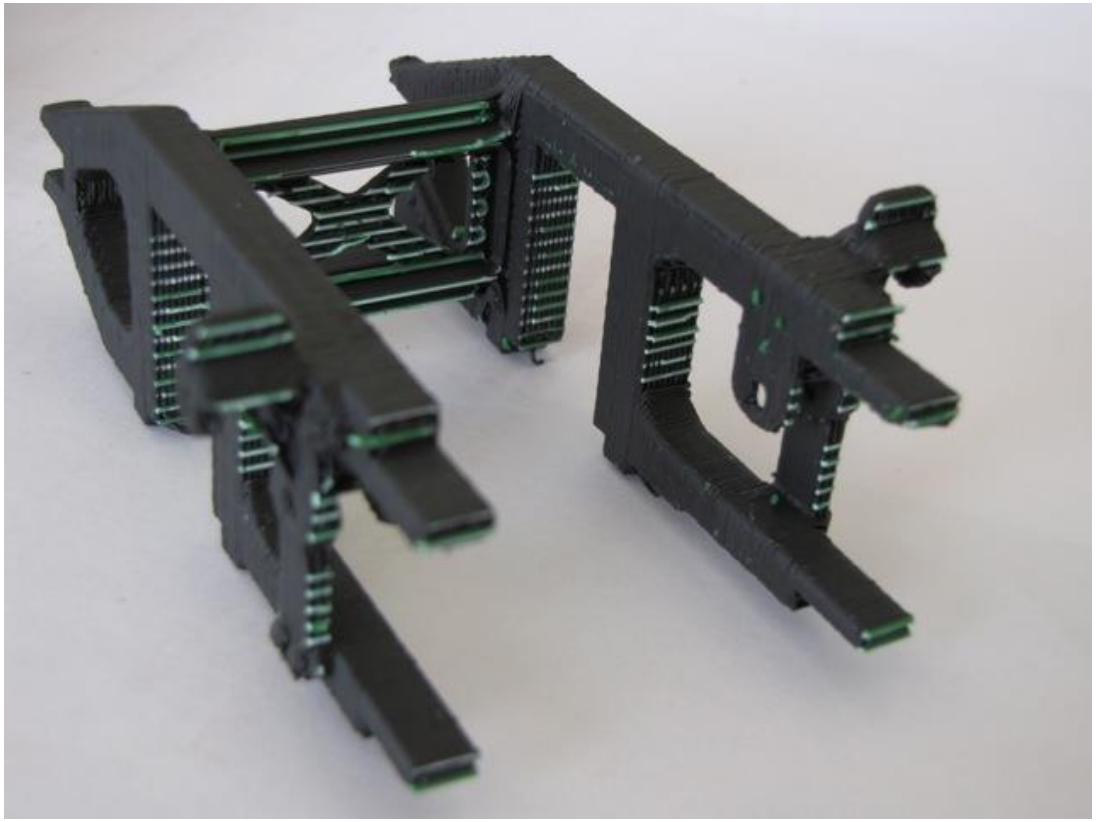
larger contact area



smaller contact area

Model Orientation: Support Contact Area

- After support removal

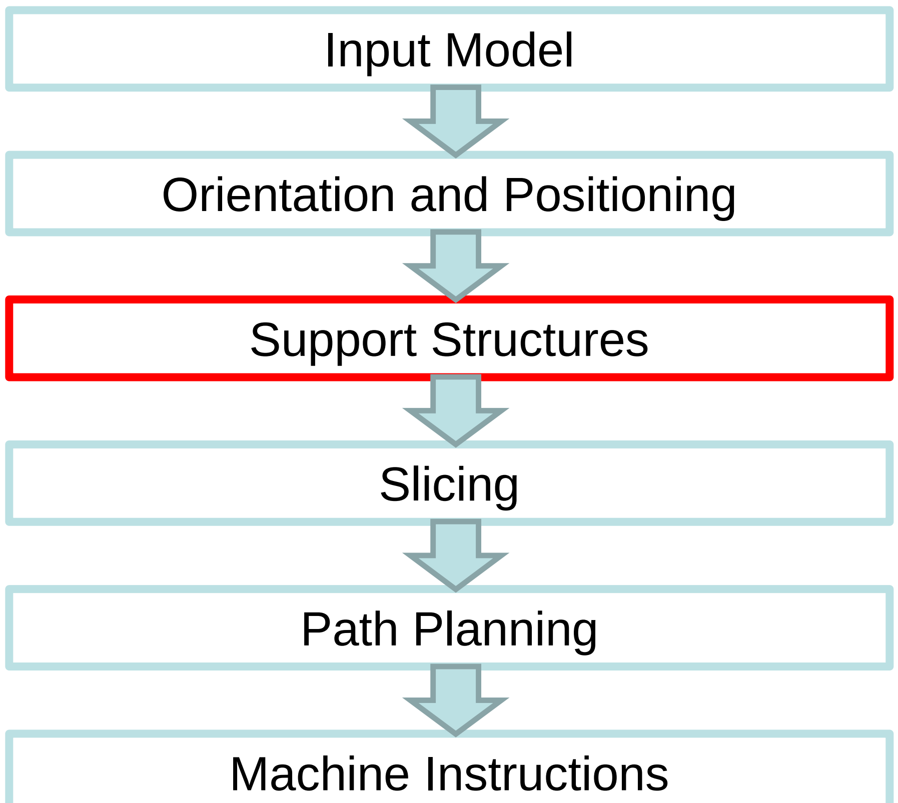


Source: http://electecnik.blogspot.com/2011/12/fds-frame-after-support-removal.html

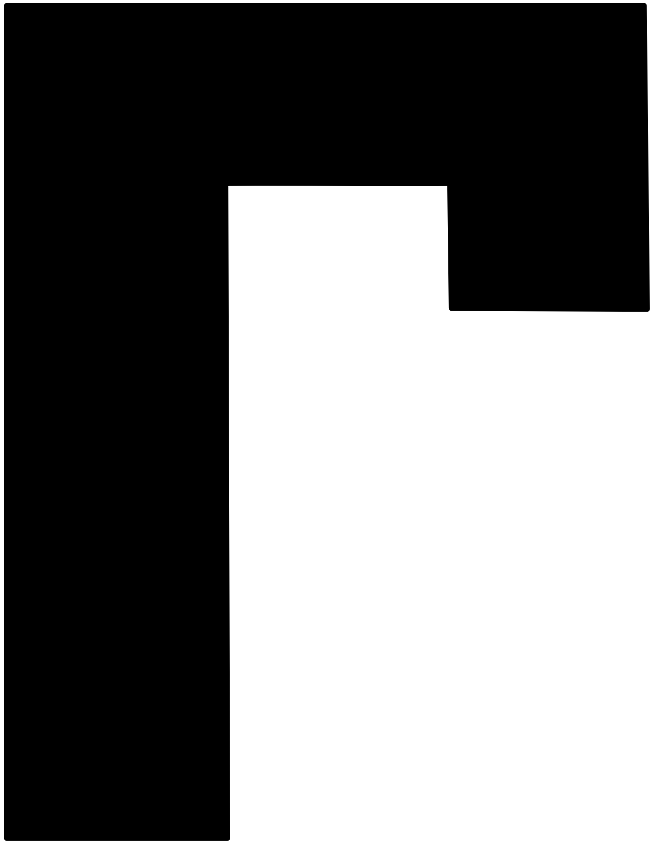
Algorithms for Specifying Model Orientation

- 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)

3D Printing Software Pipeline

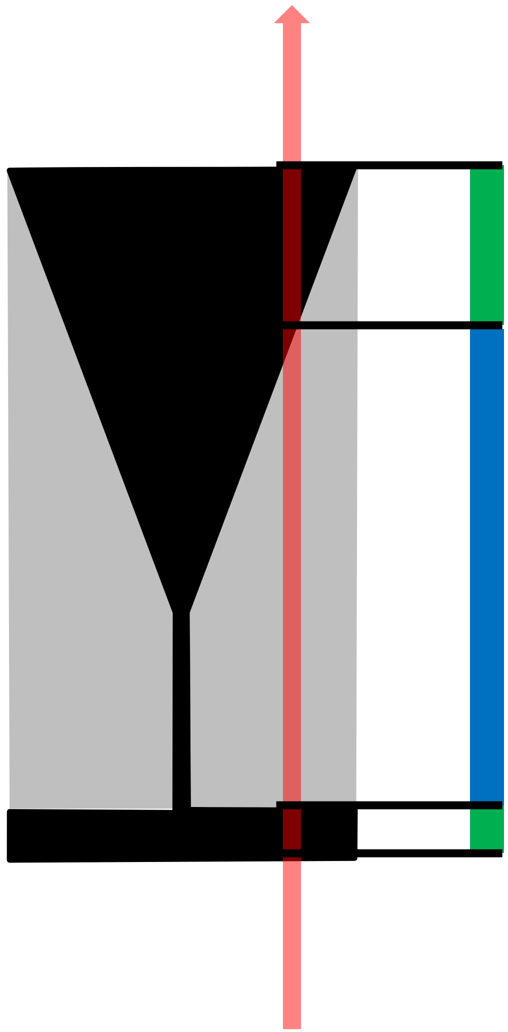


Why Support Structures?



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

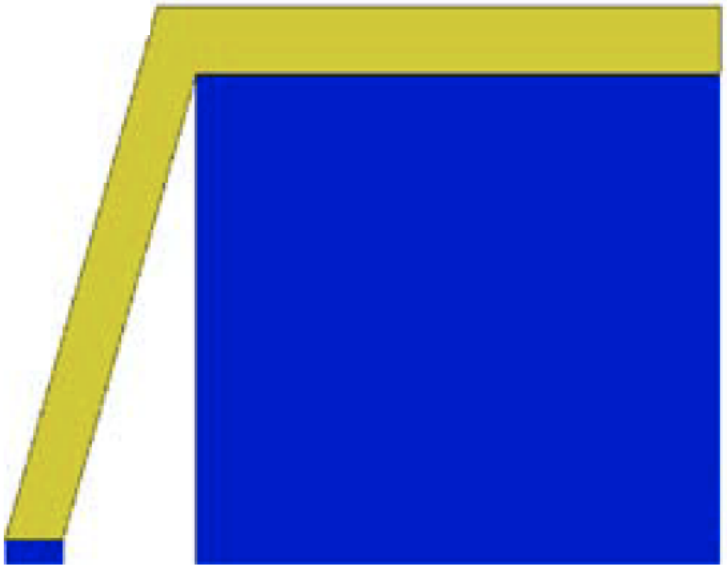


Support Generation: Draft Angle

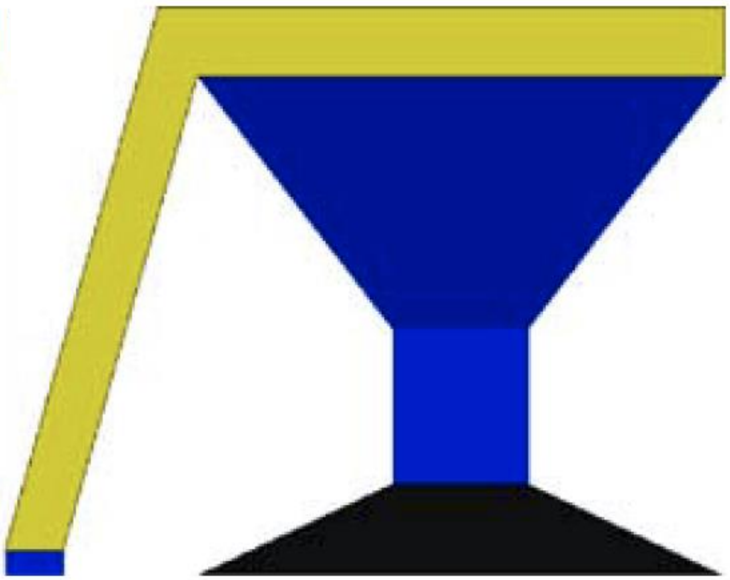
- Many printers can print at some draft angle

Support Generation: Draft Angle

- Many printers can print at some draft angle



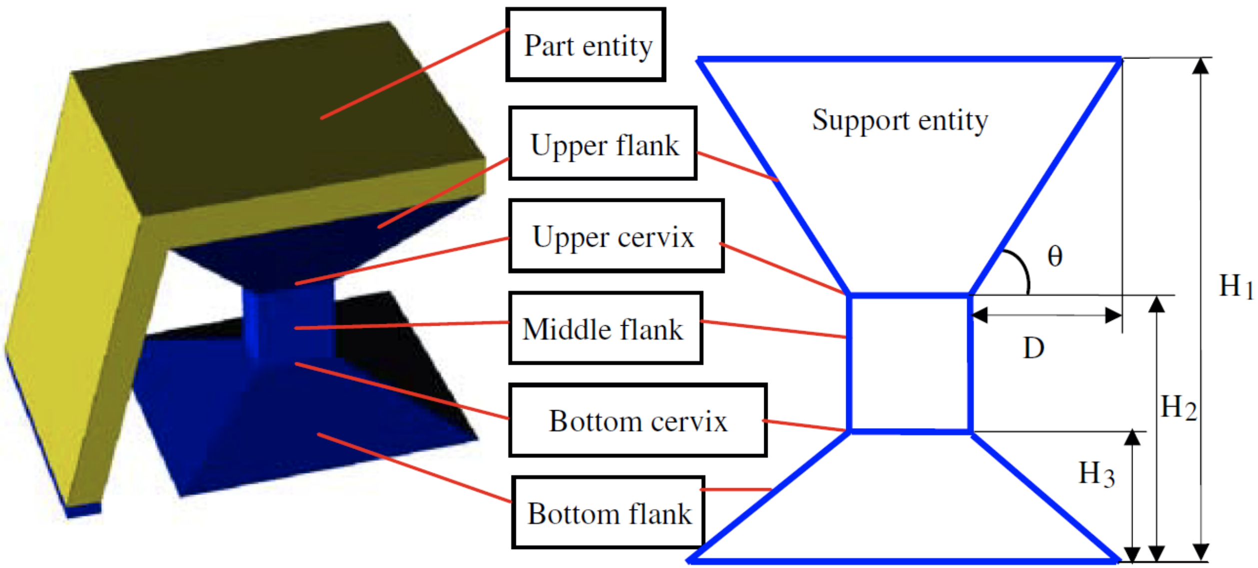
unoptimized support



optimized support

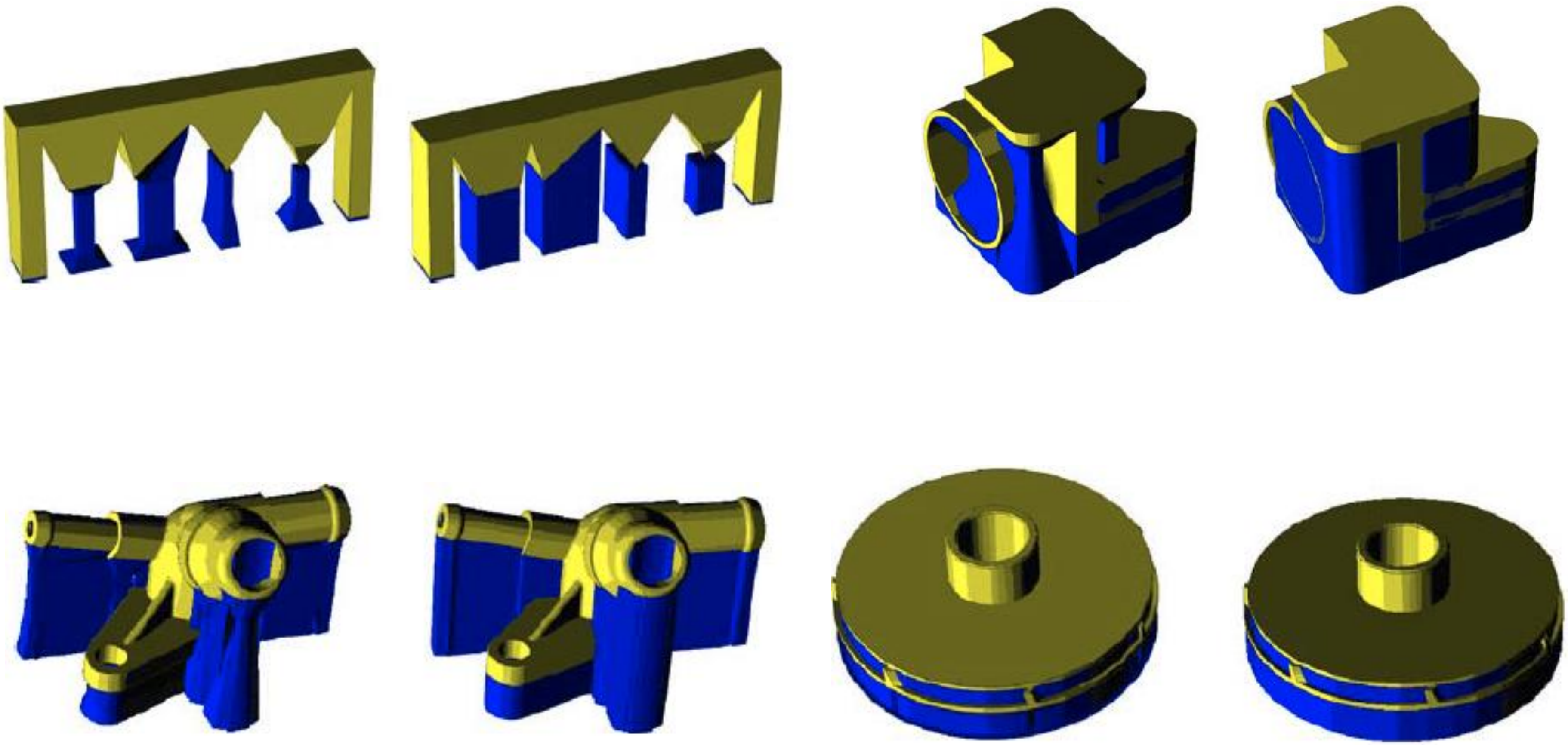
Source: Huang et al. 2009

Optimized Support Structures



Source: Huang et al. 2009

Unoptimized vs. Optimized Support Structure



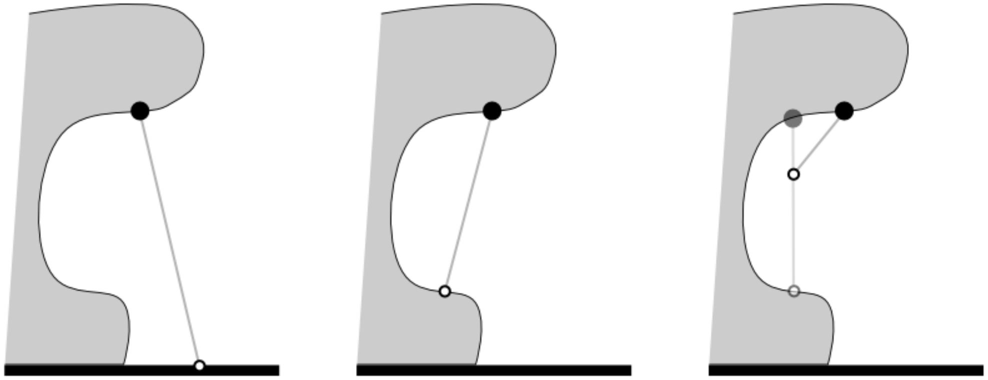
Source: Huang et al. 2009

Advanced Algorithms: MeshMixer

- Assume we have a point that needs support

- Can connect point (via support post) to:
  1) Ground
  2) Object
  3) Another post

- Problems to solve:
  1) Support Points?
  2) Connection Strategy?

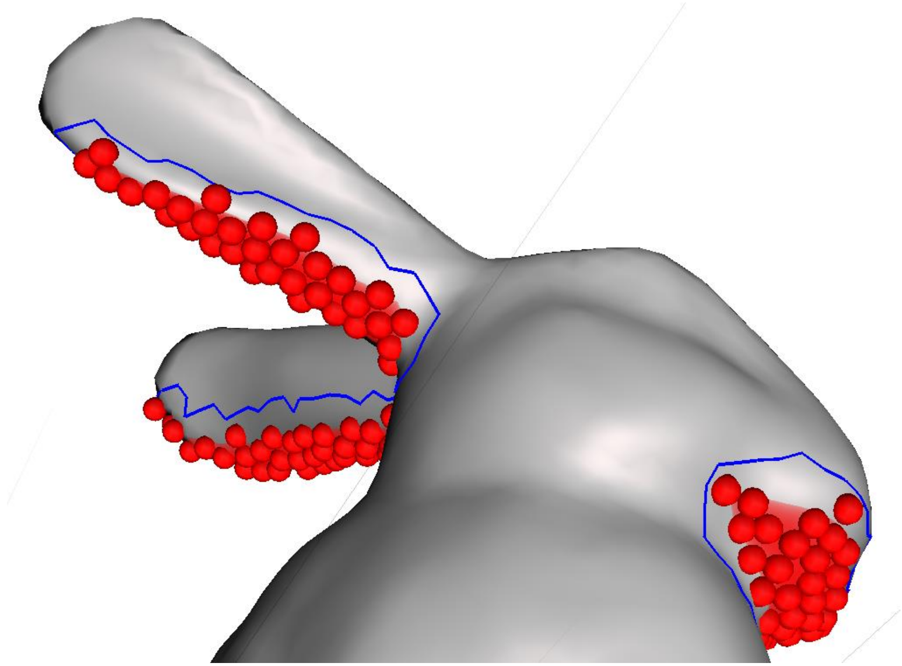


Source: Schmidt and Umetani

Advanced Algorithms: MeshMixer

- Compute point-sampling of overhang areas

- Densely cover overhangs from bottom to top

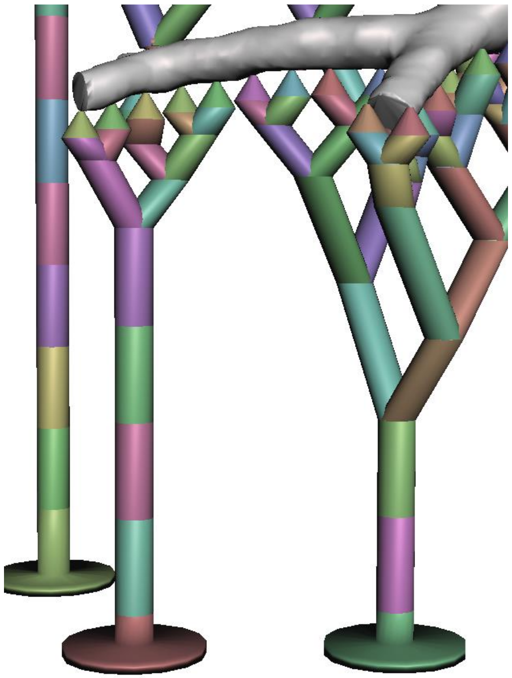


Source: Schmidt and Umetani

Advanced Algorithms: MeshMixer

Top-Down Growing Strategy

1) Add support points to priority queue
   - Sorted top-down
2) While queue is not empty
   - Pop topmost point P
   - Try to connect downwards to ground/object
   - Else try to pair with closest free point
   - Else grow downwards a fixed step.
   - Add new point to queue

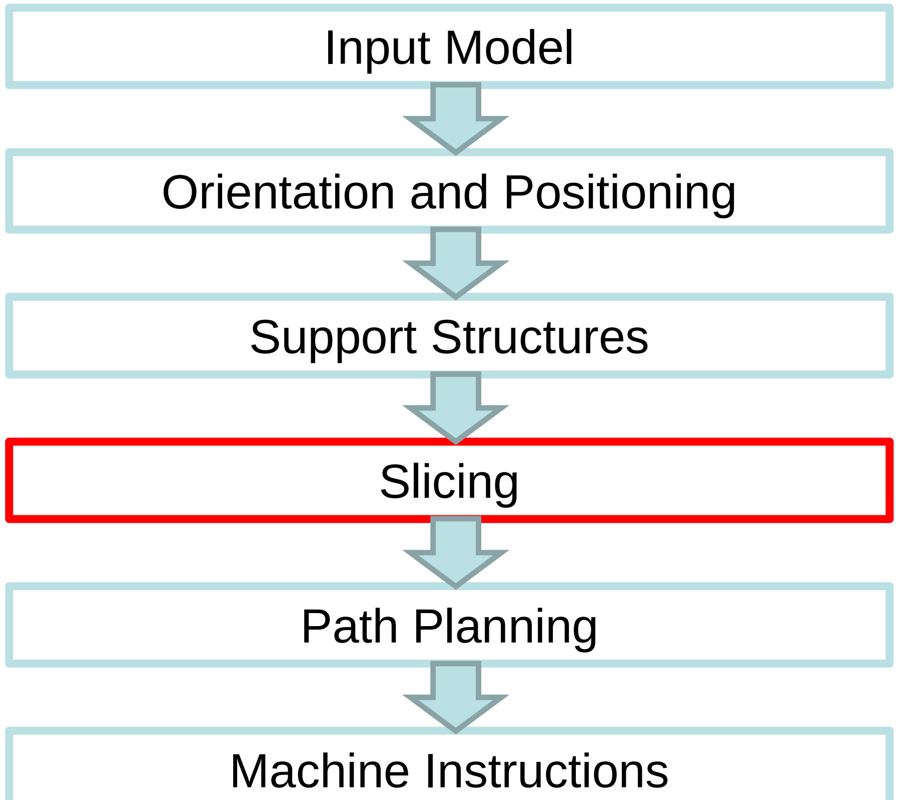


Source: Schmidt and Umetani

Advanced Algorithms: MeshMixer

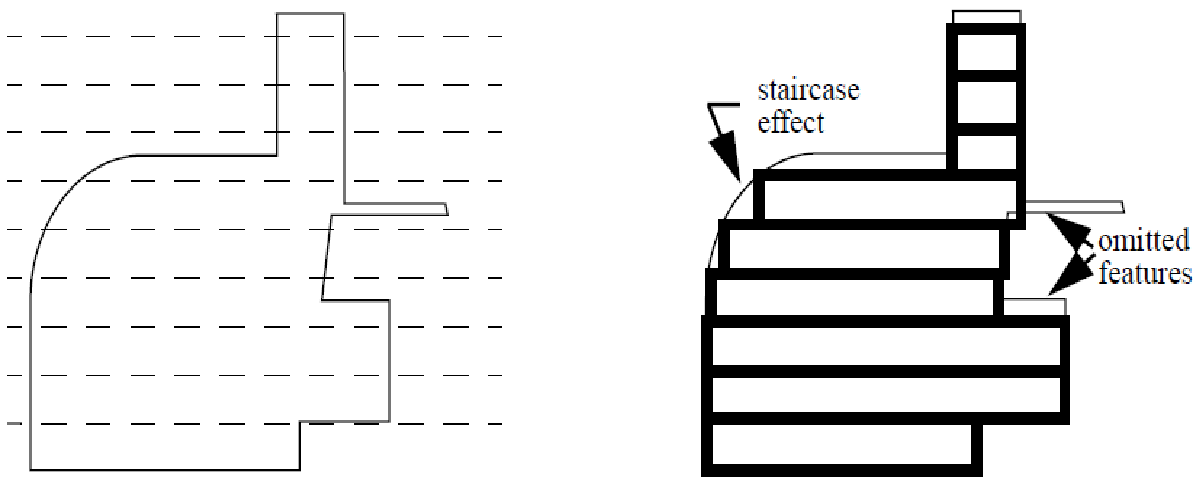
Source: http://www.youtube.com/watch?v=aFTyTV3wwsE

3D Printing Software Pipeline



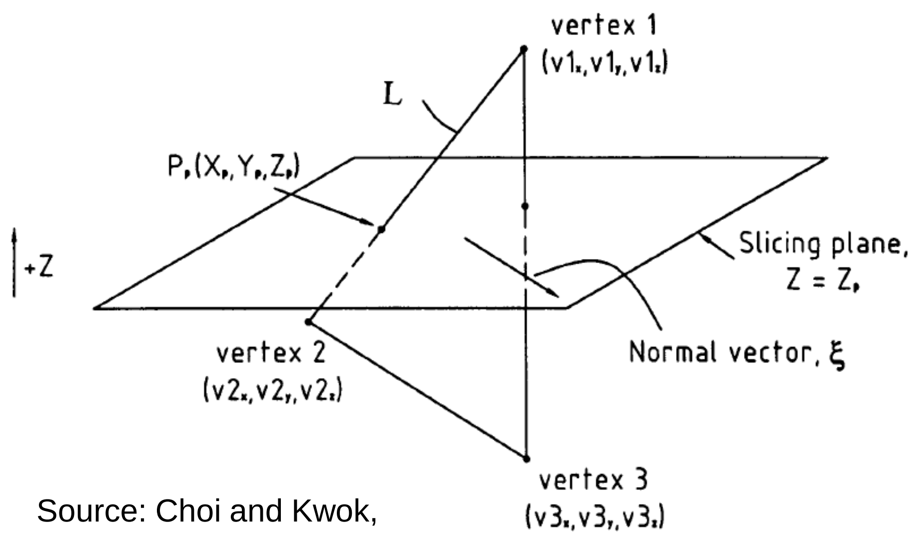
Slicing

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



Slicing Algorithms: Direct Plane-Triangle Intersection

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

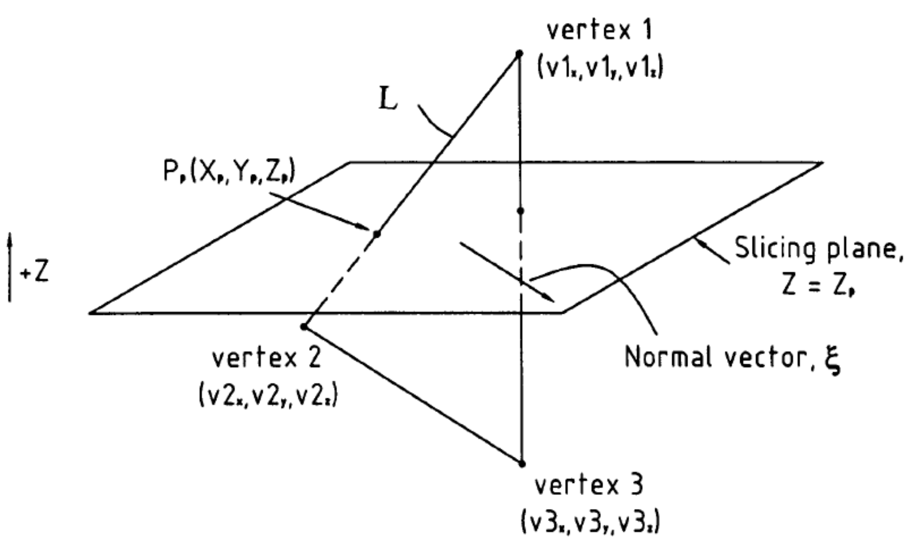


Source: Choi and Kwok, 2002

Slicing Algorithms: Direct Plane-Triangle Intersection

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

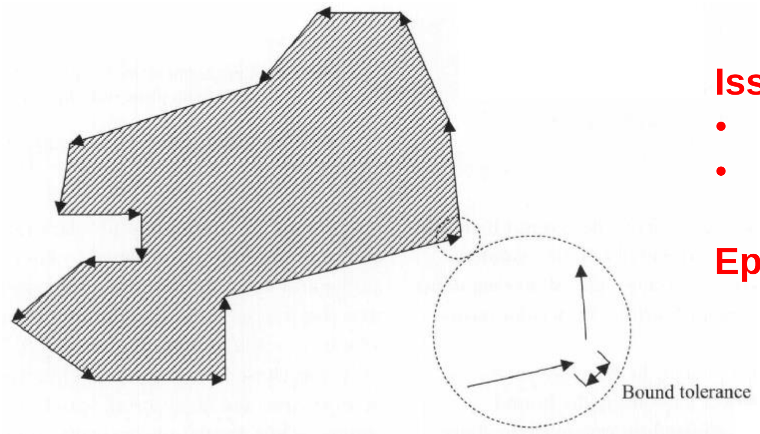
1. **Intersect each edge with the plane**
2. **If two intersection points, connect them to form a line segment**



Source: Choi and Kwok, 2002

Slicing Algorithms: Direct Plane-Triangle Intersection

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

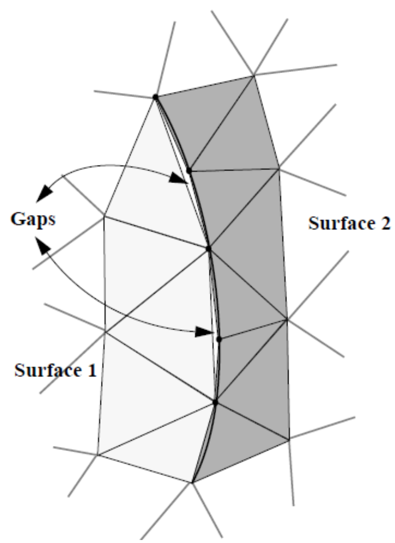


Issues
- **spurious line segments**
- **missing line segments**

Epsilons

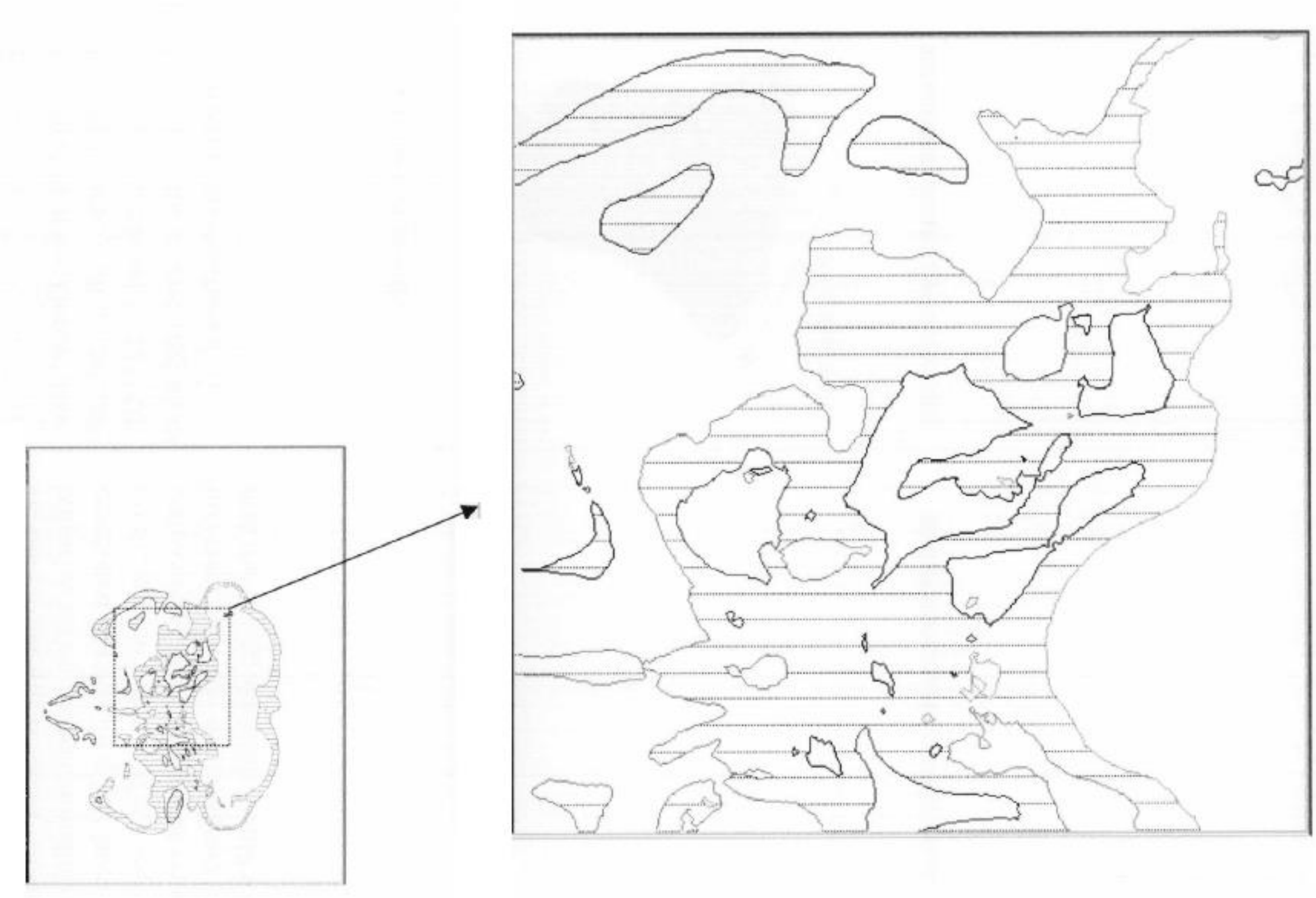
Slicing Algorithms: Direct Plane-Triangle Intersection

- STL models are not always watertight -> epsilons



Source: Marsan et al, 1998

Slicing Results with Direct Plane-Triangle Intersection



Source: Choi and Kwok, 2002

3D Printing Software Pipeline

Input Model

↓

Orientation and Positioning

↓

Support Structures

↓

Slicing

↓

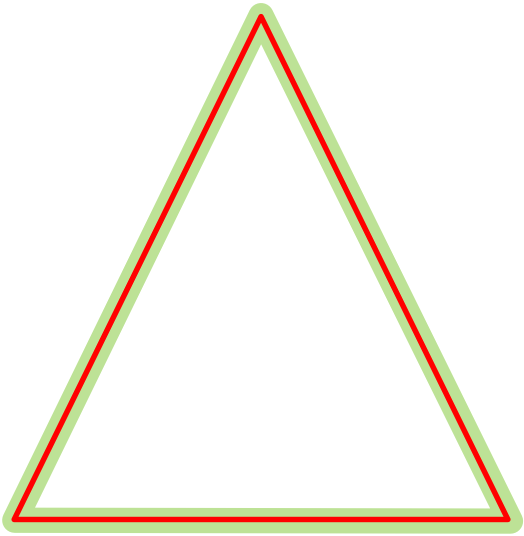
Path Planning

↓

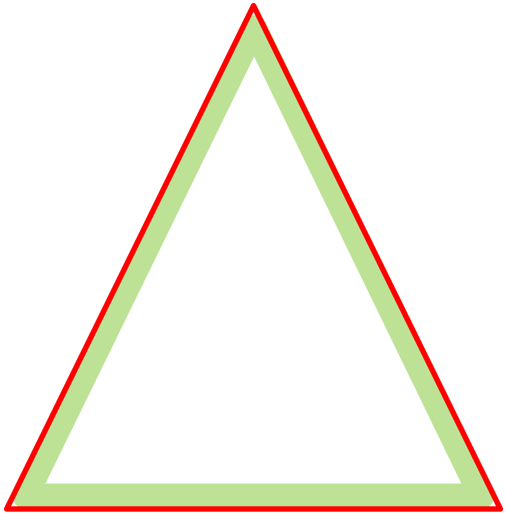
Machine Instructions

Path Planning for Vector-based 3D Printing: Contour

- Offset inwards by distance equal to the filament radius
  - offset vertices
  - remove self-intersections and model intersections



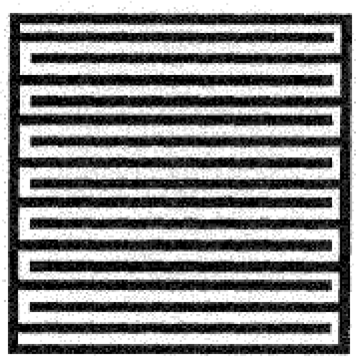
no offset



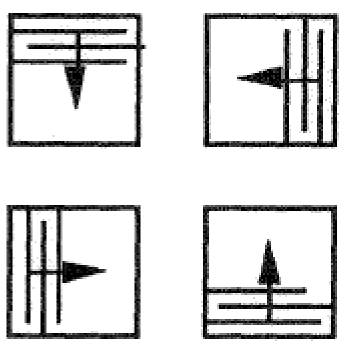
offset inwards

Path Planning for Vector-based 3D Printing: Interior

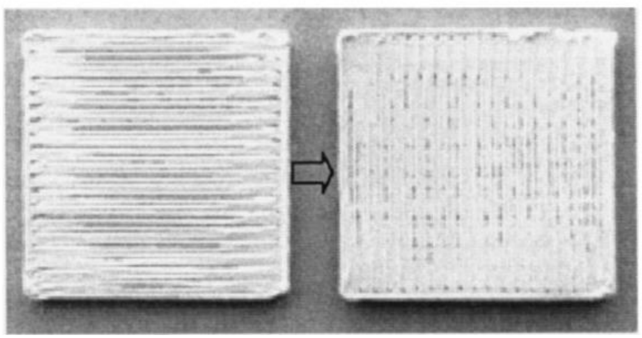
- Tracing contours is combined with filling the interior
- The interior can be completely filled



retracted hatch



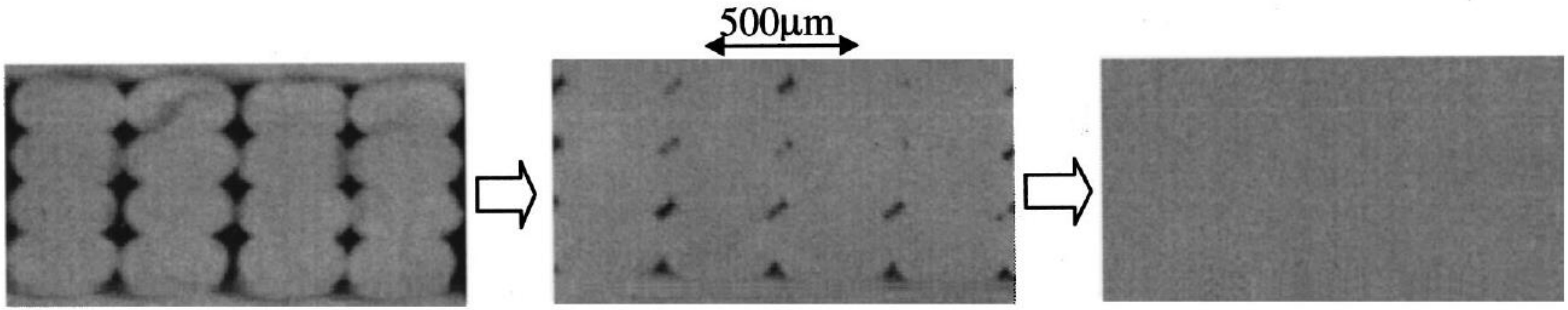
alternate sequencing



Source: Horton et al 1993

Source: Han et al 2002

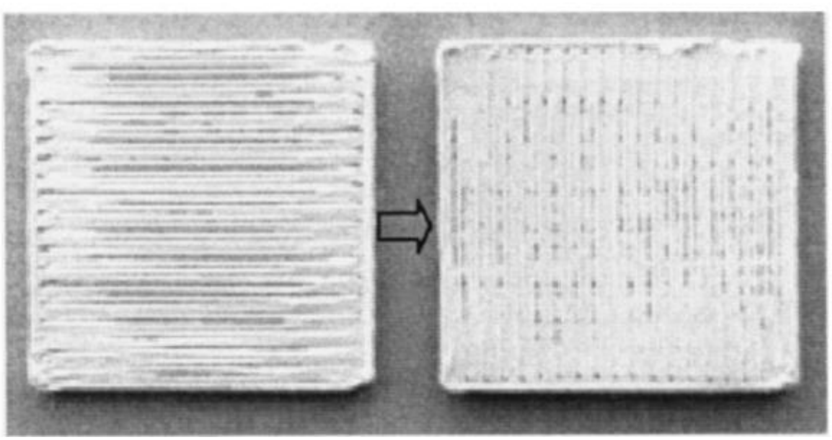
Filling in the Whole Object



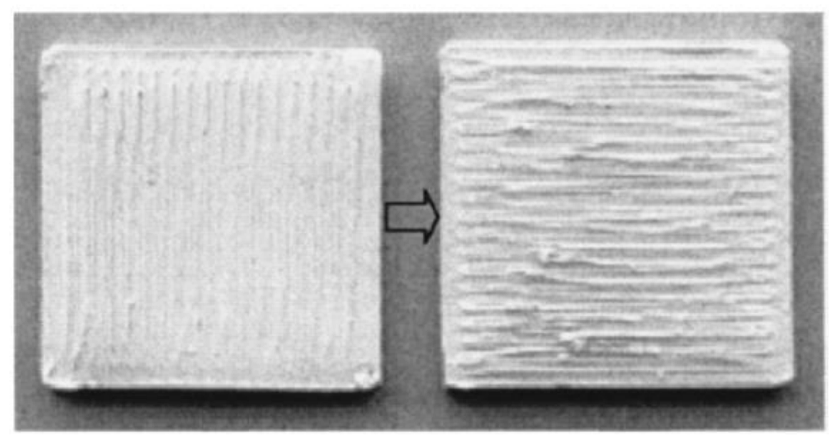
Underfills with zero offset

Underfills with insufficient negative offset (-25µm)

Fully dense fills with sufficient offset (-75µm)



underfill

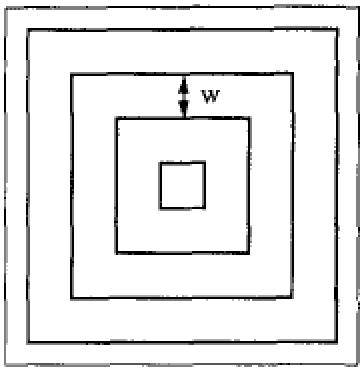


overfill

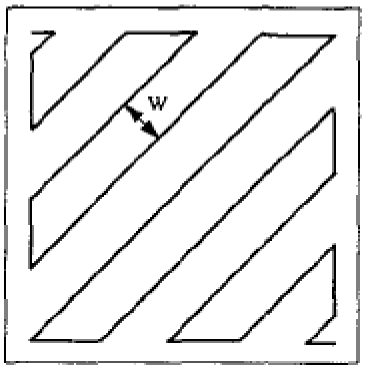
Han et al 2002

More Fill Patterns

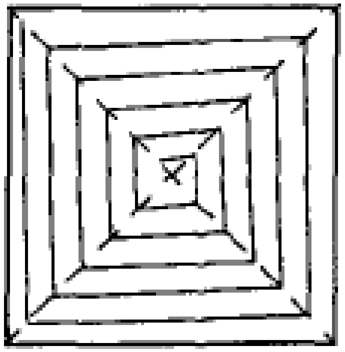
- In-plane
- Out-of-plane



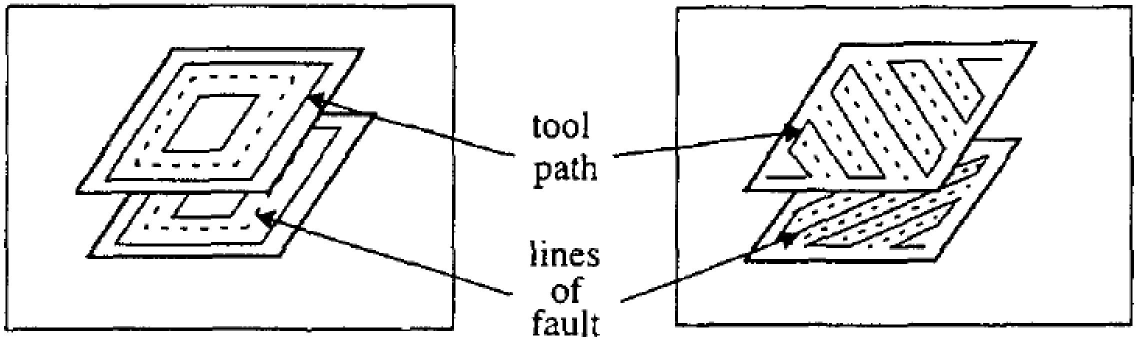
contour



raster



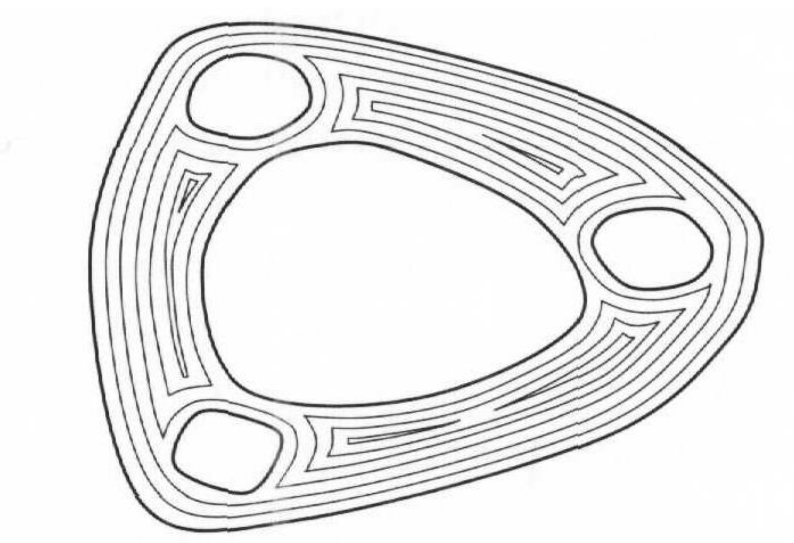
spiral

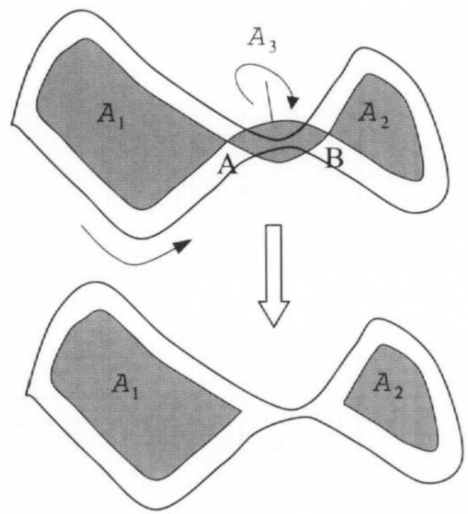


Source: Kulkarni and Dutta 1999

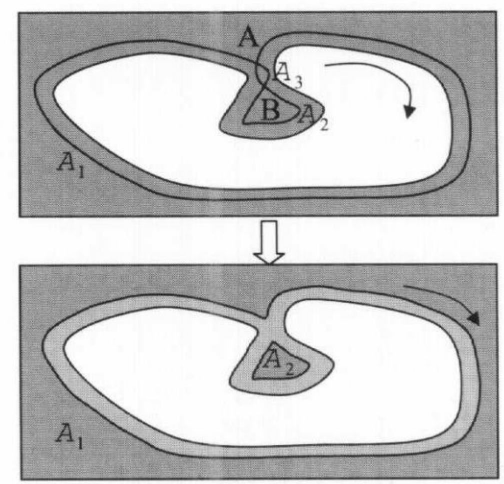
Equidistant Path Generation

- A set of paths are extracted based on distance to the surface





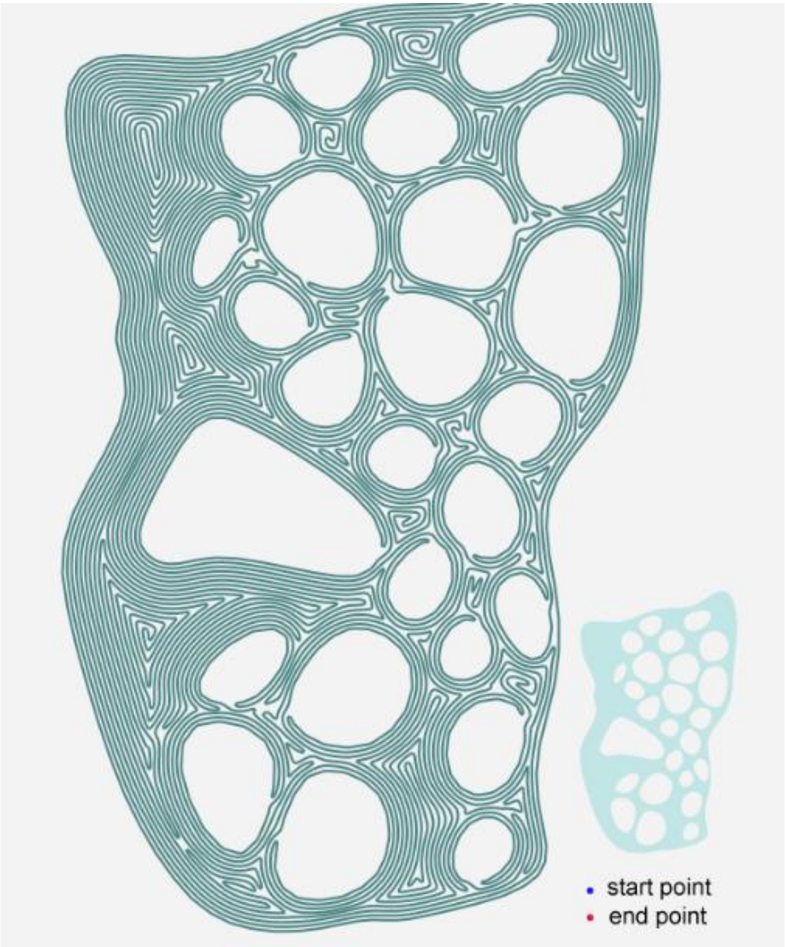
inward offset



outward offset

Source: Yang et al.2002

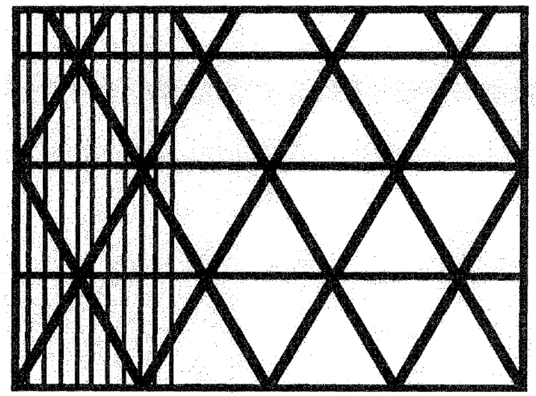
Fermat Spirals



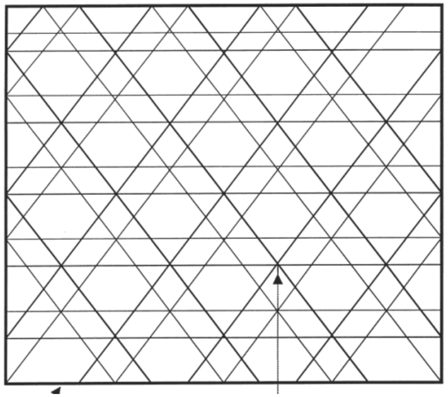
Source: ACM Siggraph

Path Planning: Interior

- Tracing contours is combined with filling the interior
- Many different fill patterns can be used



TriHatch

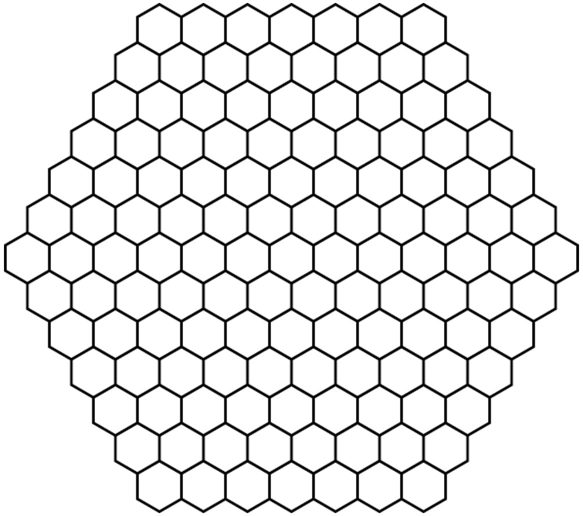


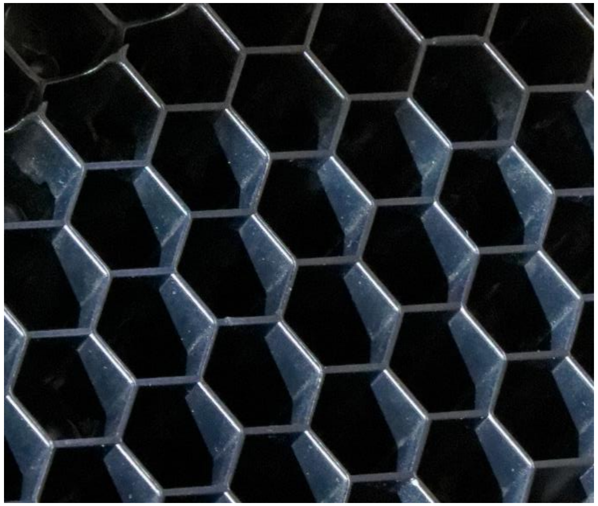
QuickCast

Source: Horton et al 1993

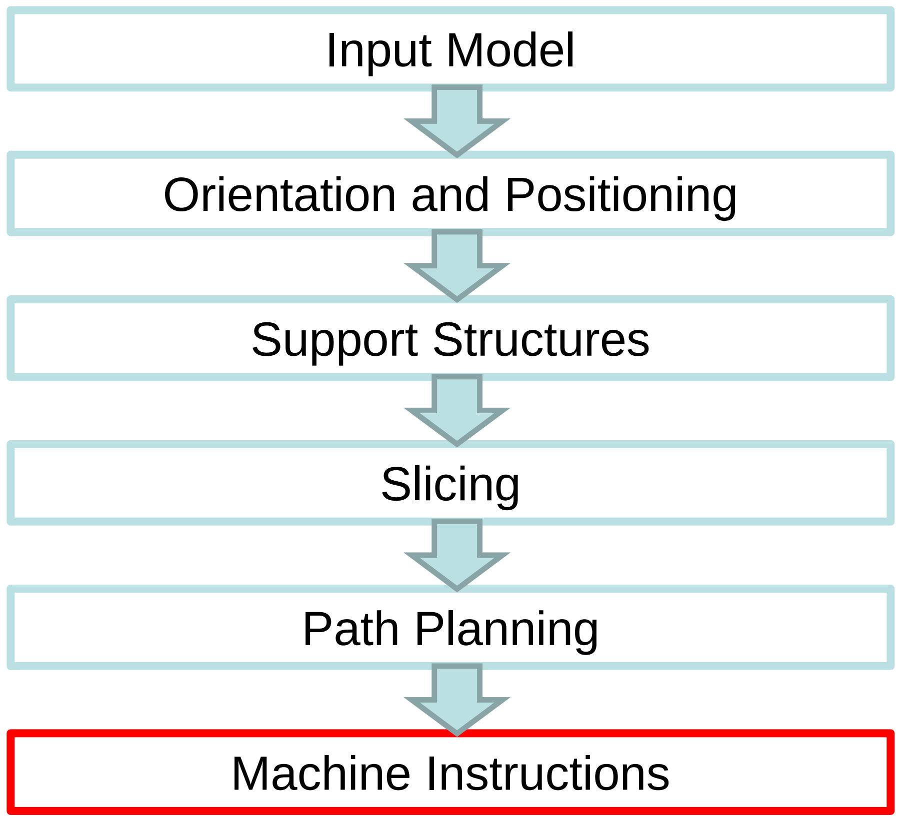
Path Planning for Vector-based 3D Printing: Interior

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





3D Printing Software Pipeline



G-code Example

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

```
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
```

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.

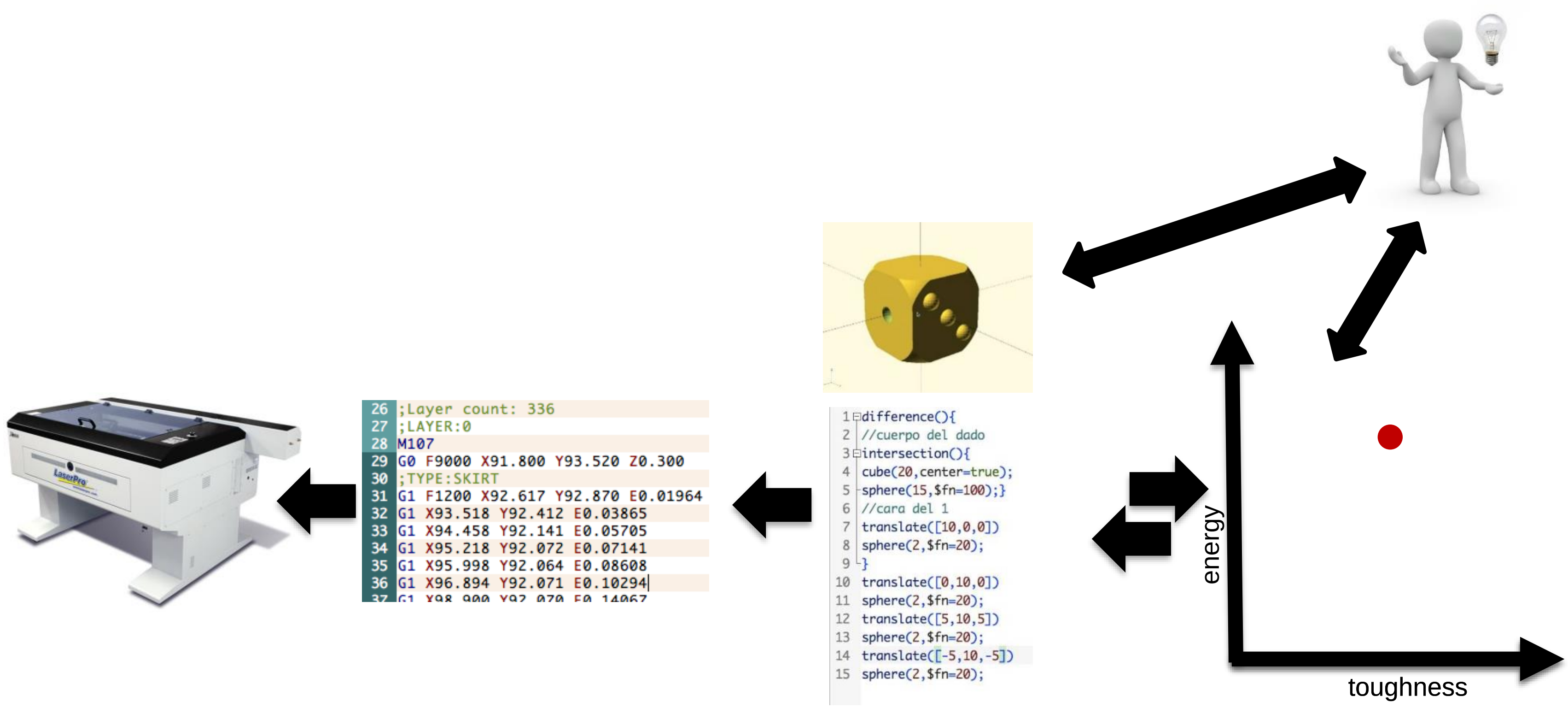
lift the Z-axis up 0.1“
seek back to X=0.0, Y=0.0, and Z=0.25

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

Course Schedule

9:00 am – 9:10 am	Introduction
9:10 am – 9:25 am	Hardware Review
9:25 am – 9:50 am	From Design to Machine Code
9:50 am – 10:20 am	Design Space Representations
10:20 am – 10:25 am	Performance Driven Design
10:25 am - 10:40 am	Break
10:40 am – 10:55 am	Performance Space Representation
10:55 am - 11:15 am	Inverse Methods
11:15 am - 11:35 am	Multi-objective Inverse Methods
11:35 am - 12:00 pm	Advanced Performance Driven Design
12:00 pm - 12:15 pm	Course Review

Computational Design Stack



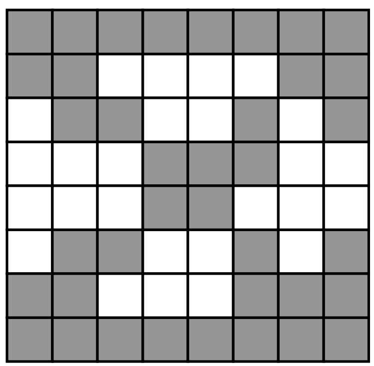
Hardware **Machine Code** **Design** **Performance**

Design vs. Design Space

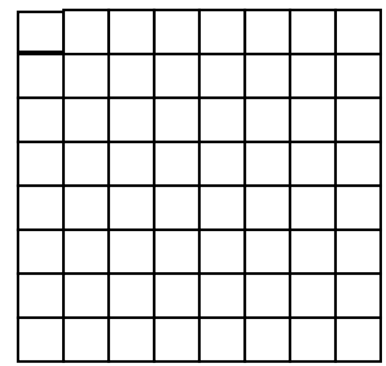
Design

Design Space

Design vs. Design Space



Design



Design Space

Low Dimensional Design Space

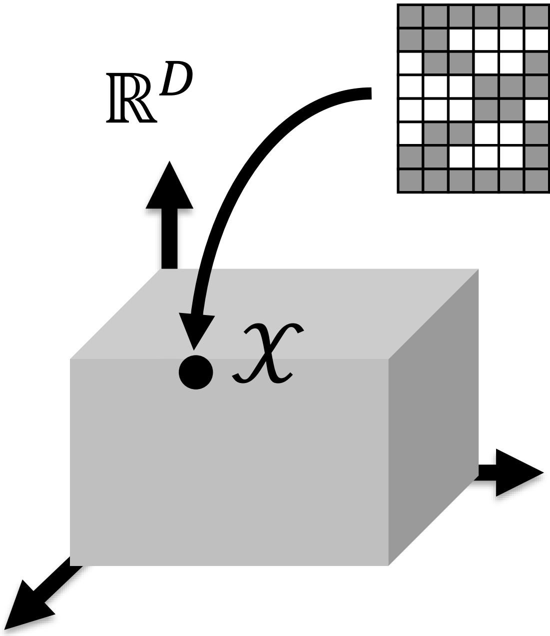
$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}$$

parameters

$\Rightarrow$

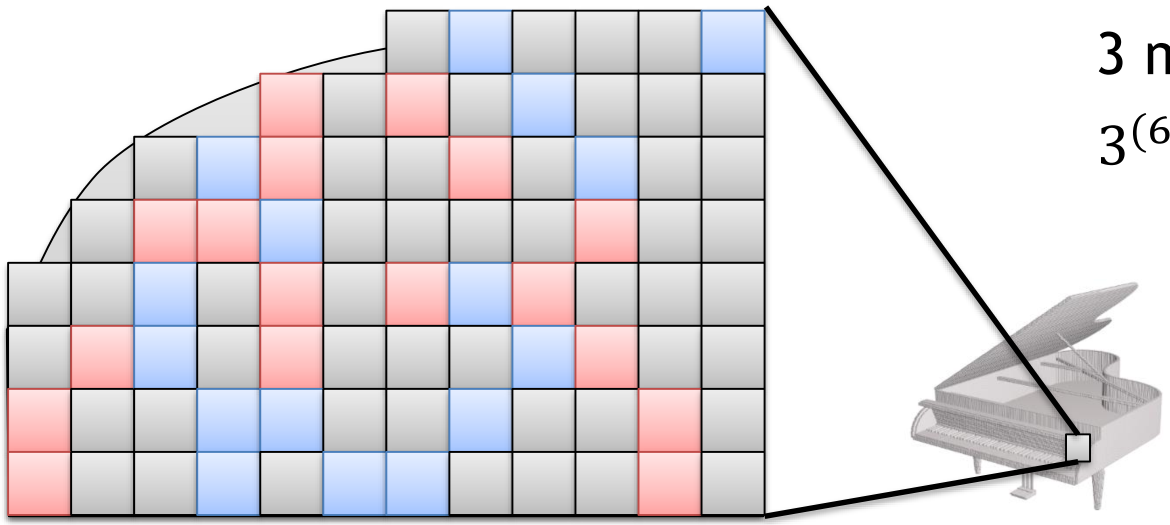
Design Space

- Each design can be mathematically represented as a point in $\mathbb{R}^D$, where D = number of voxels in a build volume



Design Space

Unconstrained Design Space



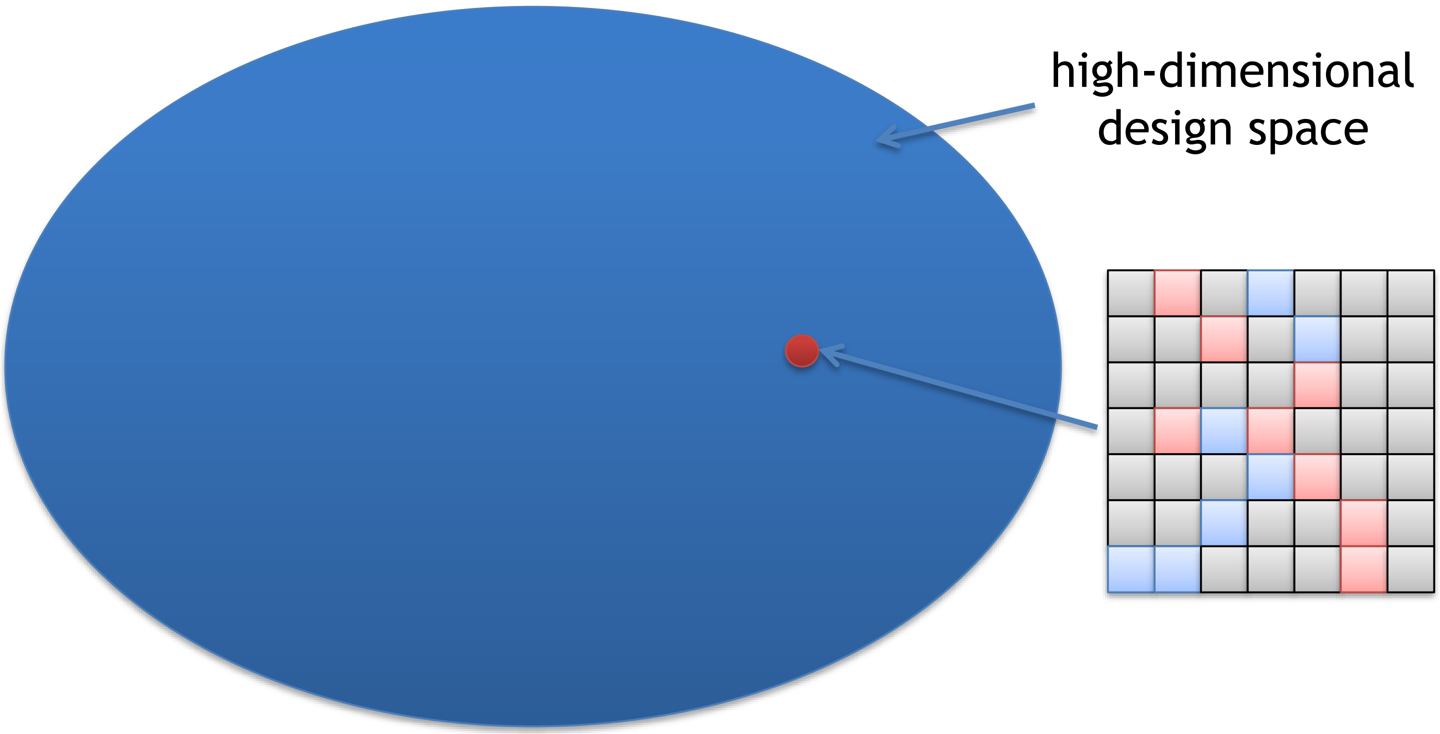
Number of combinations

1 inch cube at 600 dpi

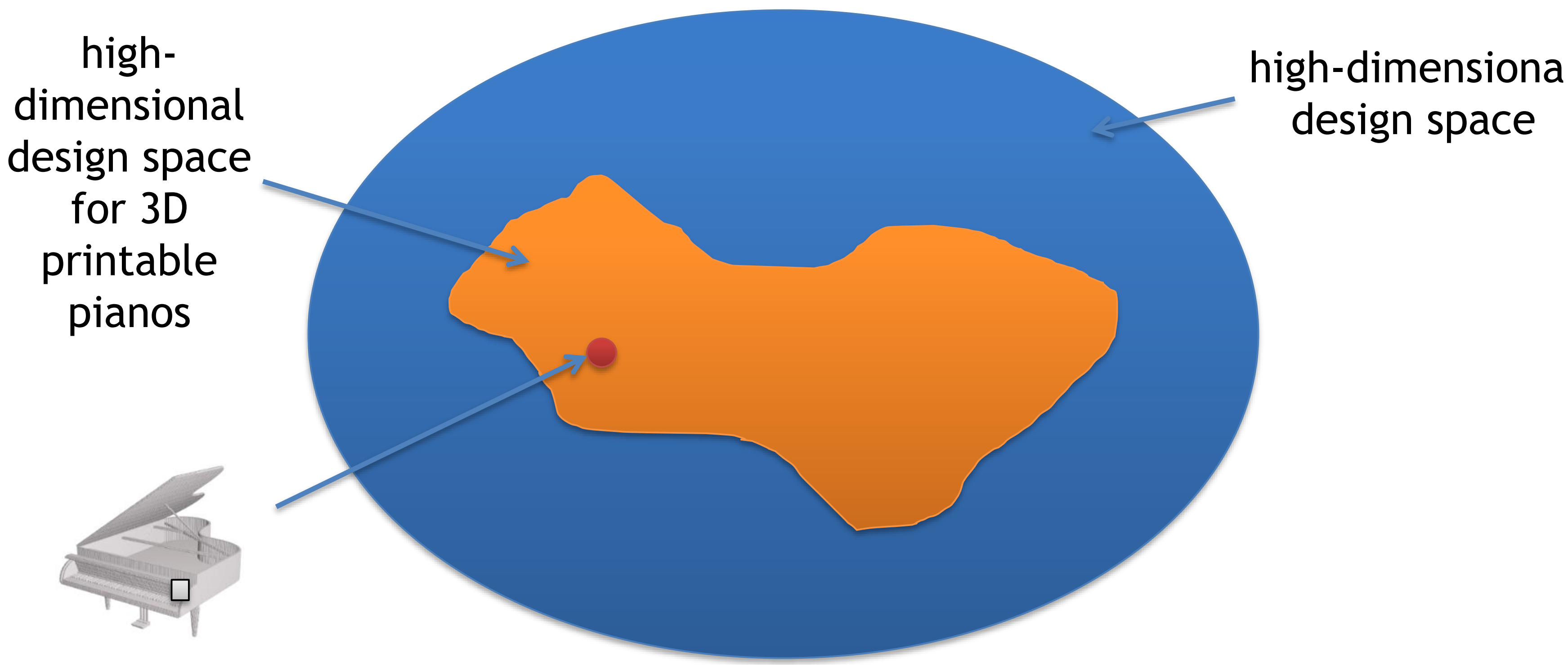
3 materials

$3^{(600^3)}$combinations

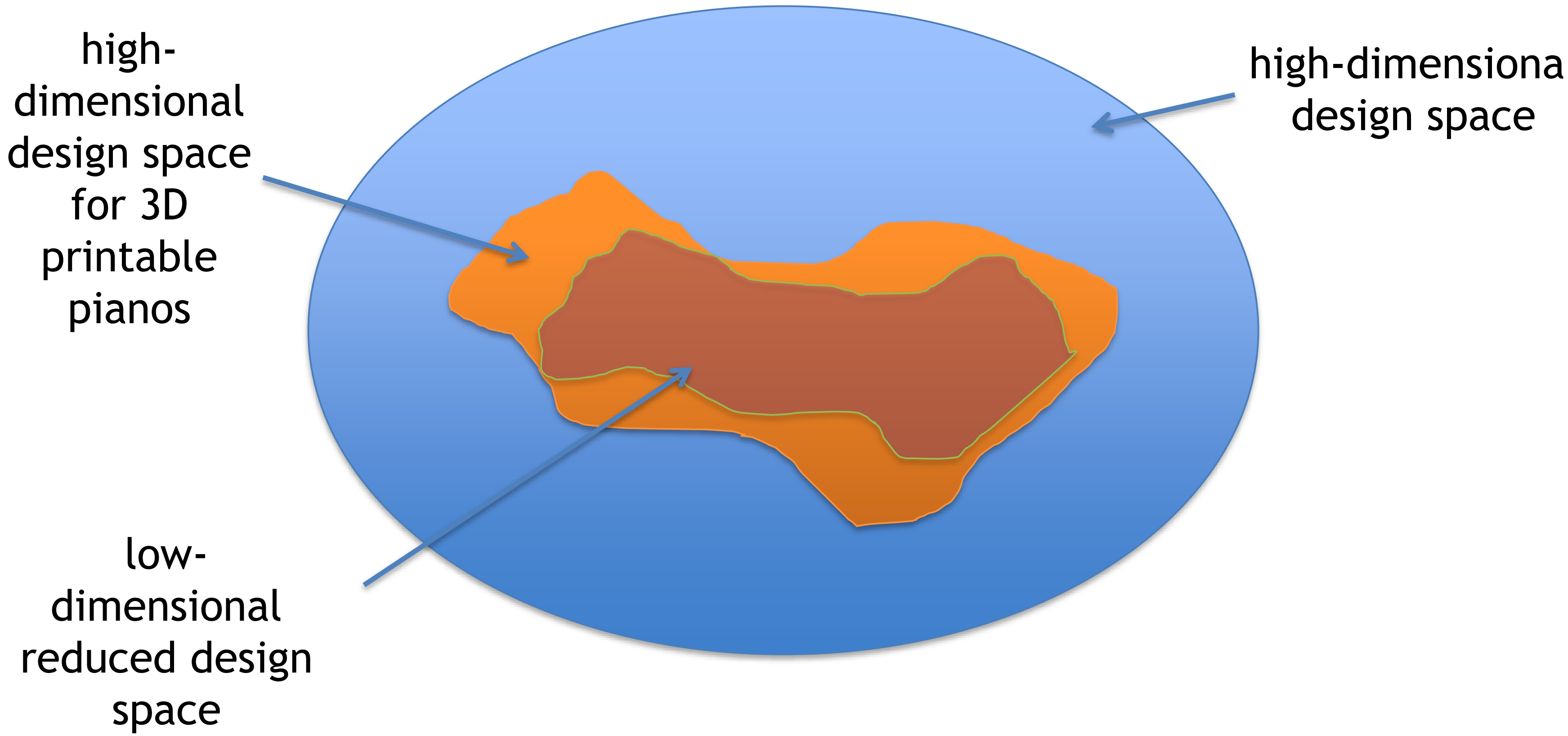
Unconstrained Design Space



The Art of Designing a Design Space

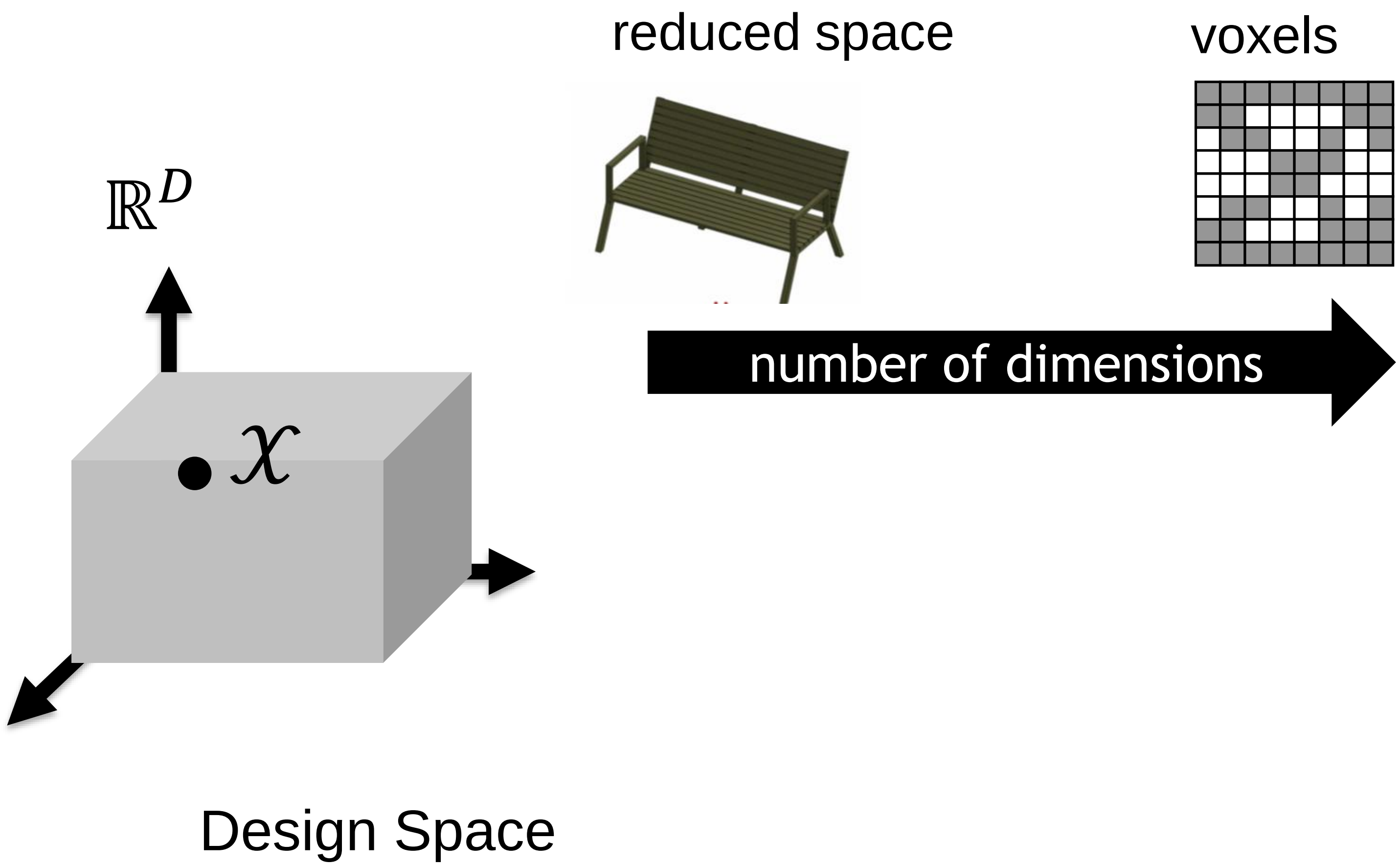


Reduced Design Spaces



Design Space Continuum

- Each design can be mathematically represented as a point in $\mathbb{R}^D$



Methods for Designing Reduced Design Spaces

- Parametric modeling
- Procedural modeling
- Deformation methods

Methods for Designing Reduced Design Spaces

- Parametric modeling
- Procedural modeling
- Deformation methods

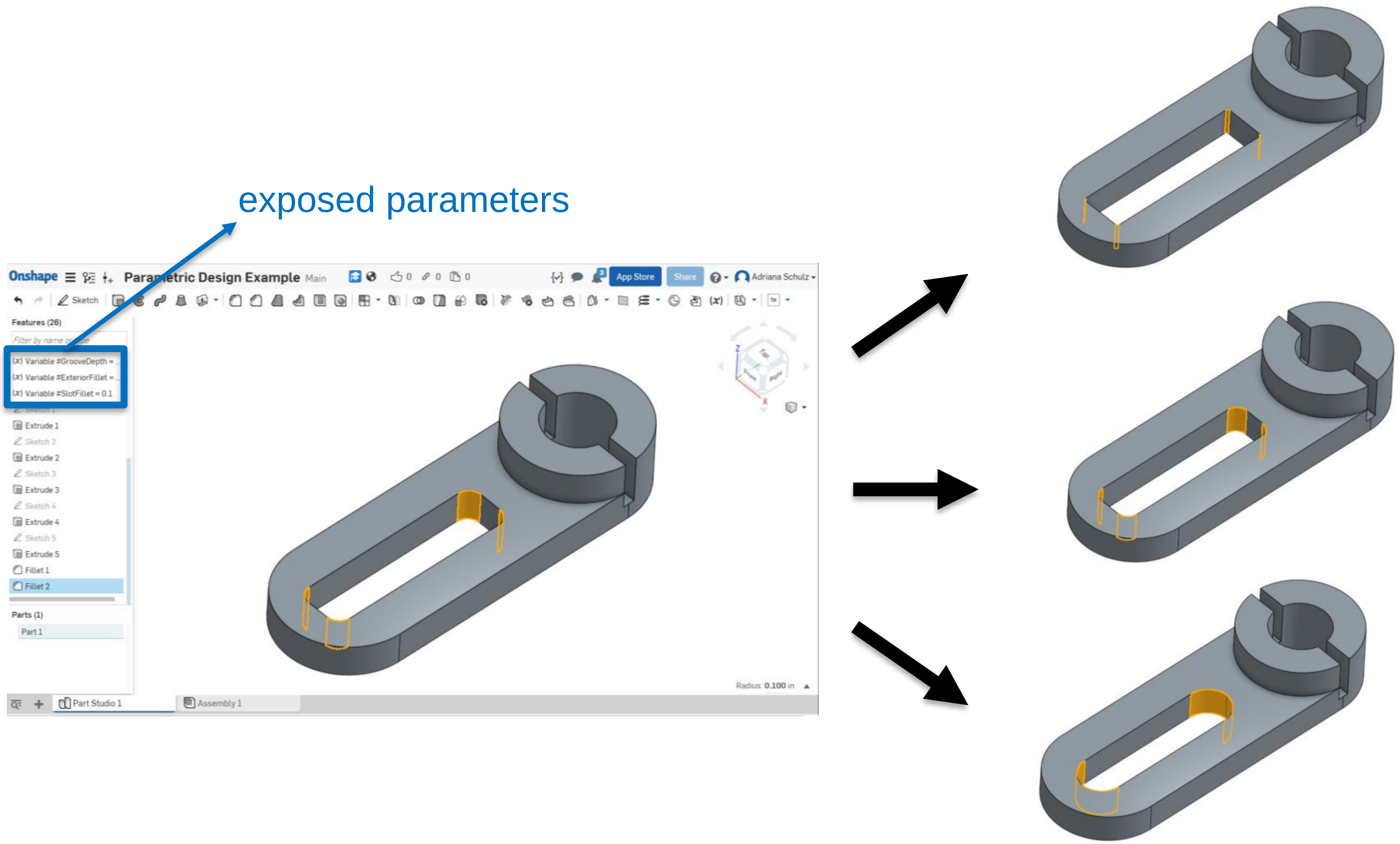
Reduced Parameters

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}$$

parameters

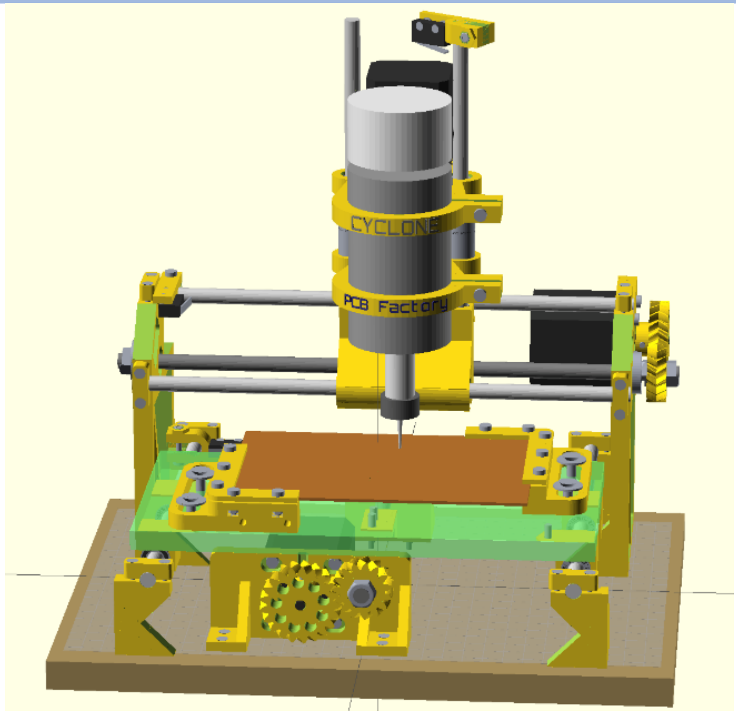
$\rightarrow$ **Parametric Design** $\rightarrow$

CAD and Expert Designed Parameters



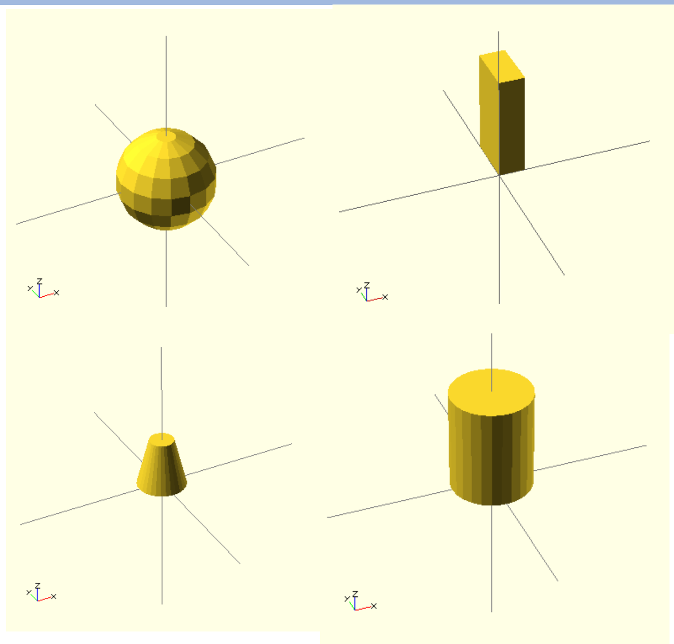
OpenSCAD

- Software for creating solid 3D CAD models
- Not an interactive modeler
  - Very basic UI
- A 3D-compiler
  - Geometry written as a script
  - Executed using CGAL/OpenCSG
  - Rendered with OpenGL
- Available for Linux/UNIX, Windows, Mac OS X
  - http://www.openscad.org



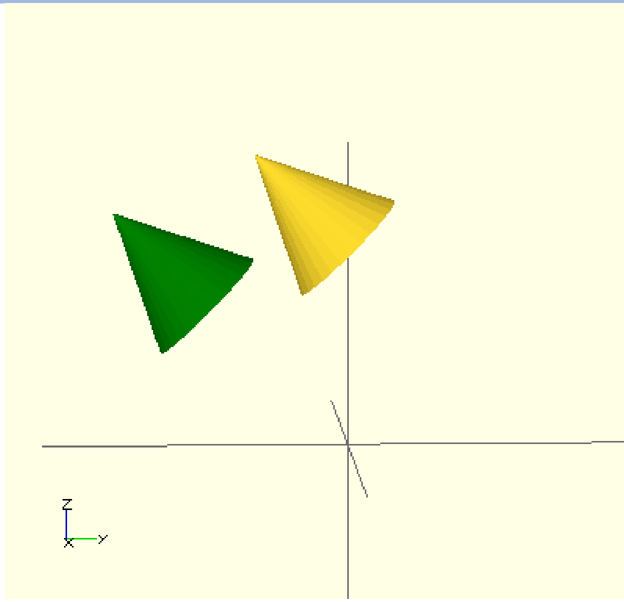
3D Primitives

- Sphere
  - `sphere(5); sphere(r=5);`
- Cube
  - `cube(5);`
  - `cube([4,8,16]);`
- Cylinder
  - `cylinder(20,10,5); cylinder(h = 20, r1 = 10, r2 = 5);`
  - `cylinder(h=20,r=10);`



Transformations

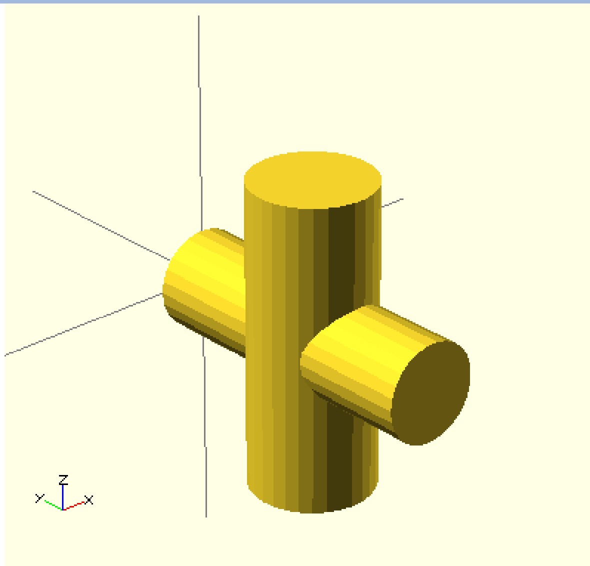
- Translate
  - e.g., `translate([10,0,0]) sphere(5); // translate along x axis`



- Rotate
- Scale
- Order dependent
  - `translate([0,0,10]) rotate([45,0,0]) cylinder([20,10,0]);`
  - `Color("green") rotate([45,0,0]) translate([0,0,10]) cylinder([20,10,0]);`

CSG

- Union
- Intersection
- Difference
- Example:

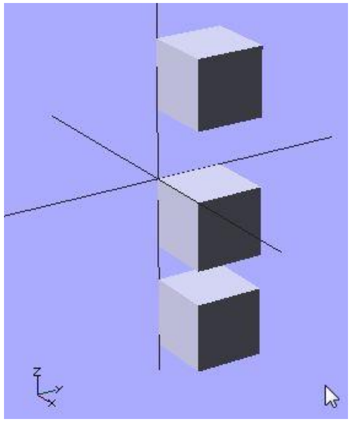



```
union()
 {
 translate([0,-25,-25]) cylinder(50,10,10);
 rotate([90,0,0]) cylinder(50,8,8);
 }
```

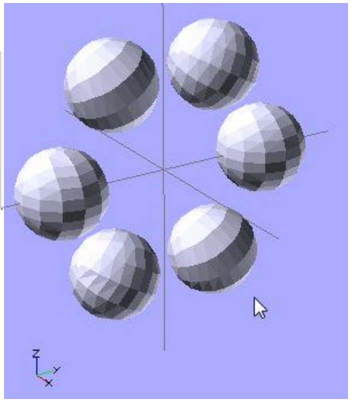
Loops

```
for (loop_variable_name = range or vector) {
    .....
    }
```

```
for ( z = [-1, 1, -2.5]) {
 translate( [0, 0, z] )
    cube(size = 1, center = false);
}
```



```
for ( i = [0:5] ) {
 rotate( i*360/6, [1, 0, 0])
  translate( [0, 10, 0] ) sphere(r = 1);
}
```



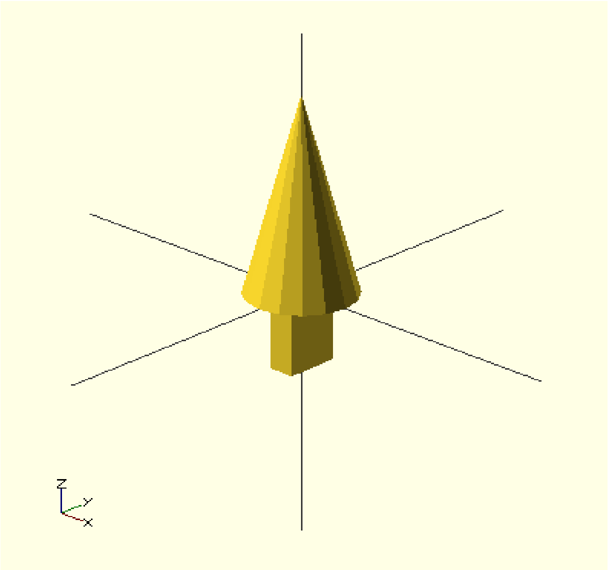
Module

- Procedures/Functions

```
module leaves() { cylinder(20,5,0); }
module box() { cube([5,10,15]); }

module tree() {
  leaves();
  scale([0.5,0.5,0.5]) translate([-2.5,-5,-15]) box();
   }

tree();
```

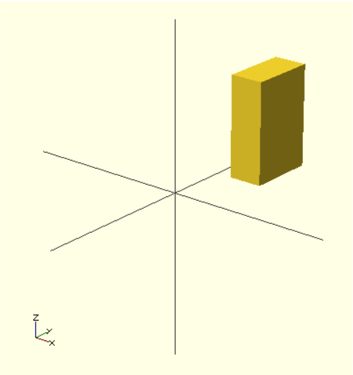


Module

- Parameters

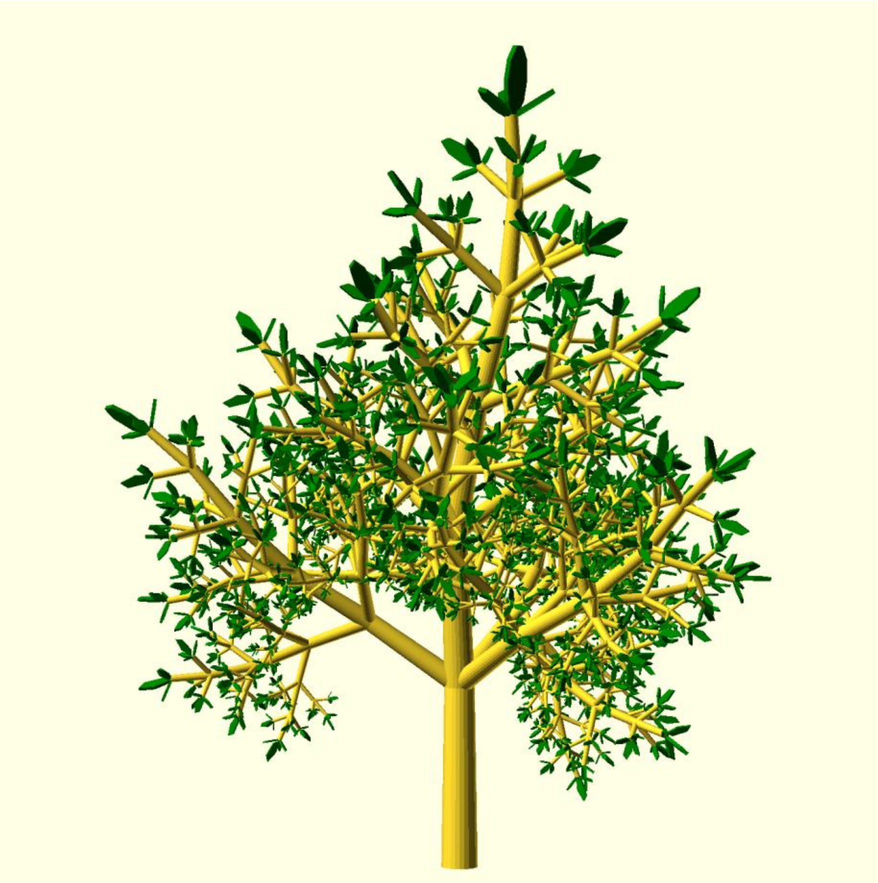
```
module box(w,l,h,tx,ty,tz)
{
        translate([tx,ty,tz])
        cube([w,l,h]);
}

box(5,10,15,10,0,5);
```



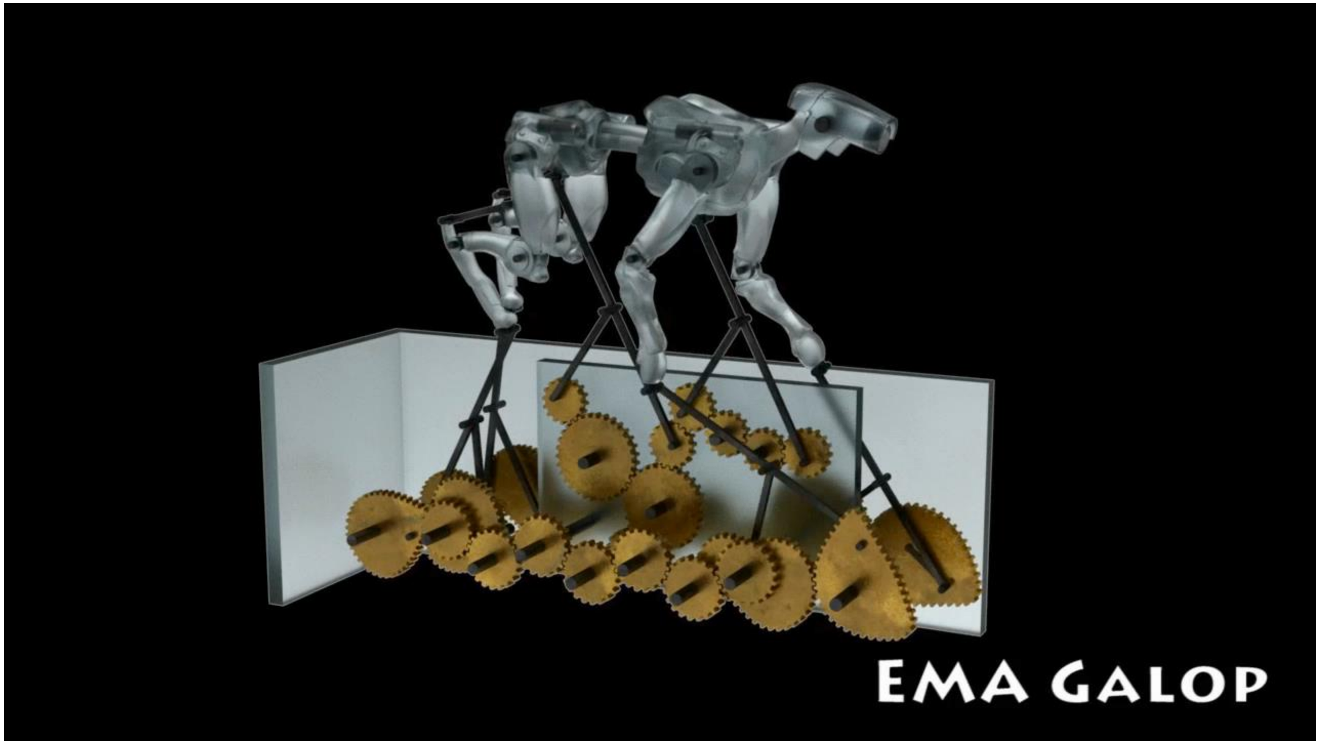
Parametric Model in OpenSCAD

```
module simple_tree(size, dna, n) {
      if (n > 0) {
         // trunk
         cylinder(r1=size/10, r2=size/12, h=size, $fn=24);
         // branches
         translate([0,0,size])
            for(bd = dna) {
               angx = bd[0];
               angz = bd[1];
               scal = bd[2];
                  rotate([angx,0,angz])
                     simple_tree(scal*size, dna, n-1);
            }
      }
      else // leaves
         color("green")
         scale([1,1,3])
            translate([0,0,size/6])
               rotate([90,0,0])
                  cylinder(r=size/6,h=size/10);
   }
   // dna is a list of branching data bd of the tree:
   //      bd[0] - inclination of the branch
   //      bd[1] - Z rotation angle of the branch
   //      bd[2] - relative scale of the branch
   dna = [ [12,  80, 0.85], [55,    0, 0.6],
         [62, 125, 0.6], [57, -125, 0.6] ];
   simple_tree(50, dna, 6);
```



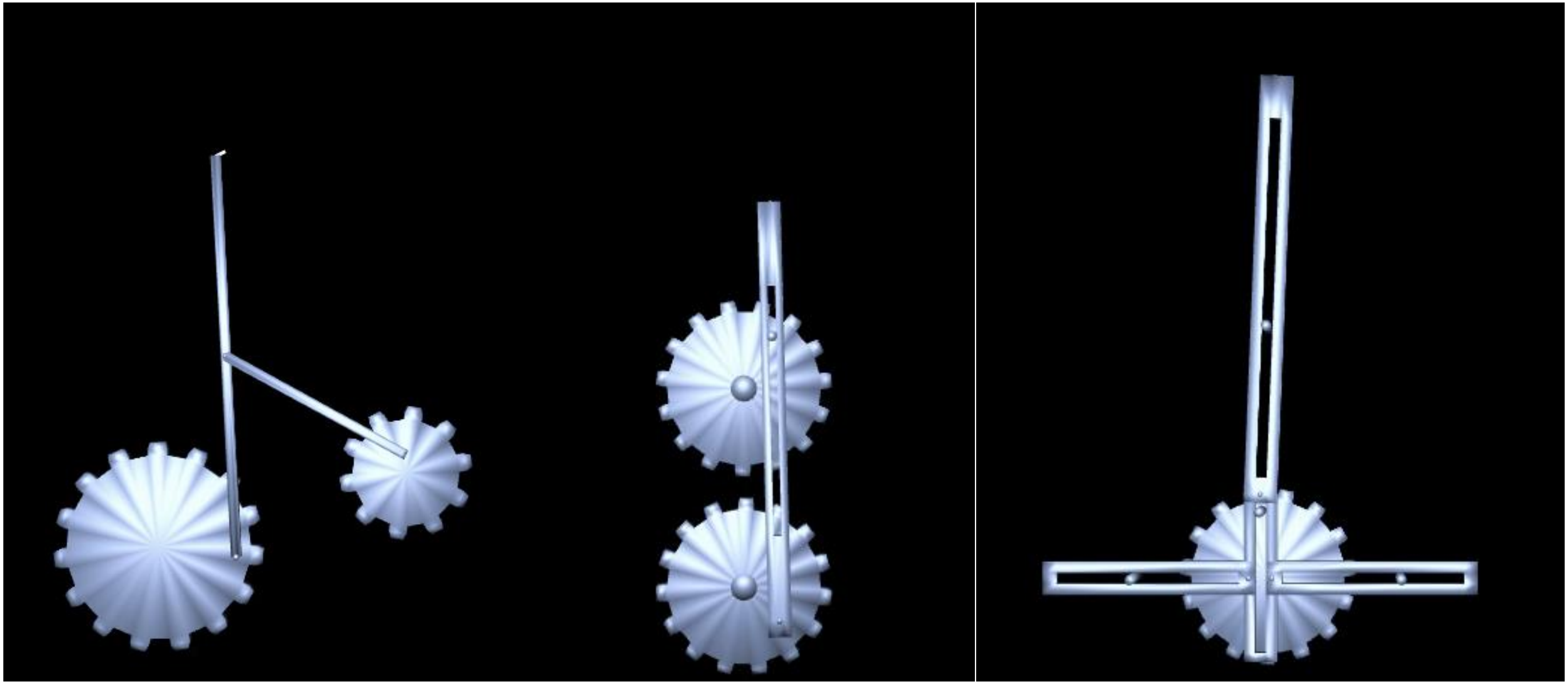
Beyond Geometry

What is the Design Space of Mechanisms?



Coros et al 2013

Library of Mechanisms



Coros et al 2013

Methods for Designing Reduced Design Spaces

- Parametric modeling
- Procedural modeling
- Deformation methods

Formal Grammars and Languages

- A finite set of **nonterminal symbols: {S, A, B}**
- A finite set of **terminal symbols: {a, b}**
- A finite set of **production rules: S $\rightarrow$ AB**
- A **start symbol: S**

- Generates a set of finite-length sequences of symbols by recursively applying production rules starting with **S**

L-systems

- **nonterminals** : 0, 1
- **terminals** : [,], +, -
- **start** : 0
- **rules** : (1 → 11), (0 → 1[+0][-0])

- **Visual representation: turtle graphics**
  - **0: draw a line segment ending in a leaf**
  - **1: draw a line segment**
  - **- turn left 45 degrees**
  - **+ turn right 45 degrees**
  - **[: push position and angle,**
  - **]: pop position and angle,**

Iterations:

0

1[+0][-0]

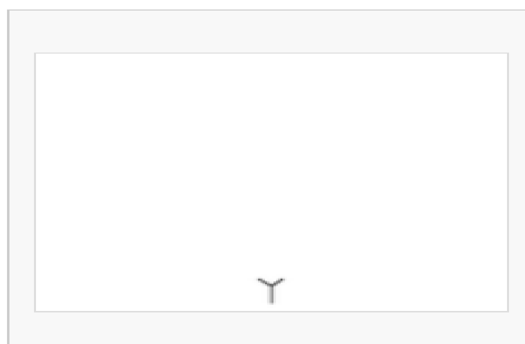
11[+1[+0][-0]][-1[+0][-0]]

L-systems

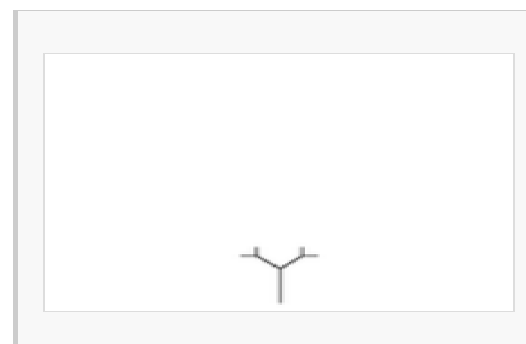
- **nonterminals** : 0, 1
- **terminals** : [,], +, -
- **start** : 0
- **rules** : ($1 \rightarrow 11$), ($0 \rightarrow 1[+0][-0]$)

- **Visual representation: turtle graphics**
  - **0: draw a line segment ending in a leaf**
  - **1: draw a line segment**
  - **- turn left 45 degrees**
  - **+ turn right 45 degrees**
  - **[: push position and angle,**
  - **]: pop position and angle,**

Axiom



First recursion



Second recursion

$n = 0$

$n = 1 : 1[+0][-0]$

$n = 2 : 11[+1[+0][-0]][-1[+0][-0]]$

L-systems

a
n=5, δ=25.7°
F
F → F[+F]F[-F]F

b
n=5, δ=20°
F
F → F[+F]F[-F][F]

c
n=4, δ=22.5°
F
F → FF-[-F+F+F]+
[+F-F-F]

d
n=7, δ=20°
X
X → F[+X]F[-X]+X
F → FF

e
n=7, δ=25.7°
X
X → F[+X][-X]FX
F → FF

f
n=5, δ=22.5°
X
X → F-[[X]+X]+F[+FX]-X
F → FF

Types of L-Systems

- *Deterministic*: If there is exactly one production for each symbol

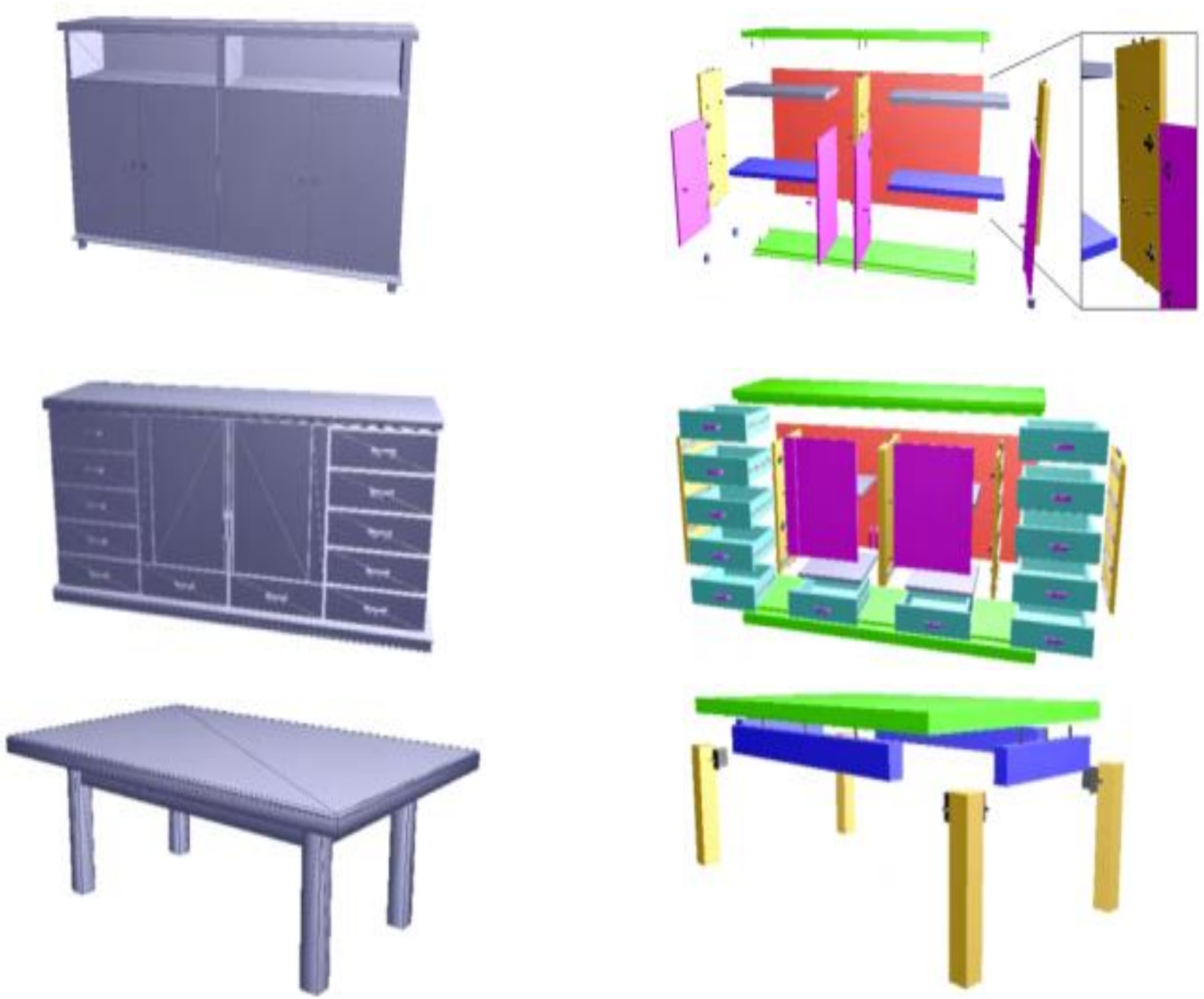
  $0 \rightarrow 1[0]0$

- *Stochastic*: If there are several, and each is chosen with a certain probability during each iteration

  $0\ (0.5) \rightarrow 1[0]0$

  $0\ (0.5) \rightarrow 0$

Furniture Design using Formal Grammar



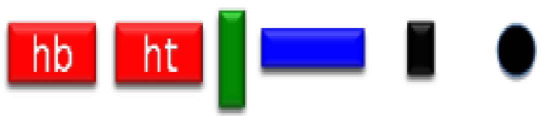
Source: Converting 3D Furniture Models to Fabricable Parts and Connectors, Lau et al., Siggraph 2011

Formal Grammar for 2D Cabinets

$$N = \{\boxed{S}, \boxed{B}, \boxed{X}, \boxed{Y}\}$$

Non-terminal Symbols
- Collection of Parts

$$\Sigma = \{hb, ht, v, ha, leg, wheel\}$$



Terminal Symbols
- Separate Parts

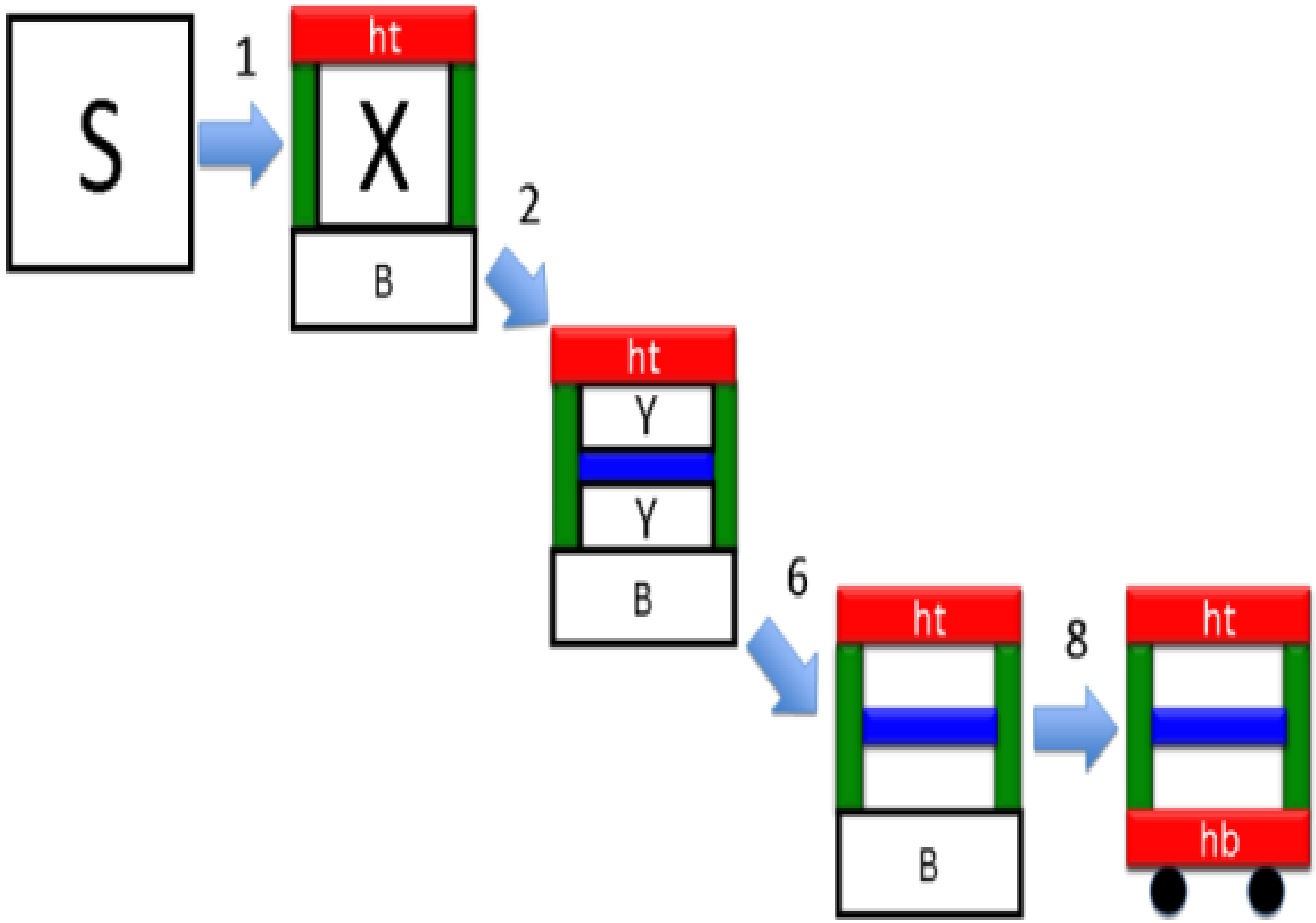
P : Set of Production Rules

$\boxed{S}$: Start Symbol

The language specifies a directed graph,
and each graph represents parts and connectors

Source: Converting 3D Furniture Models to Fabricable Parts and Connectors, Lau et al., Siggraph 2011

Sequence of Production Rules

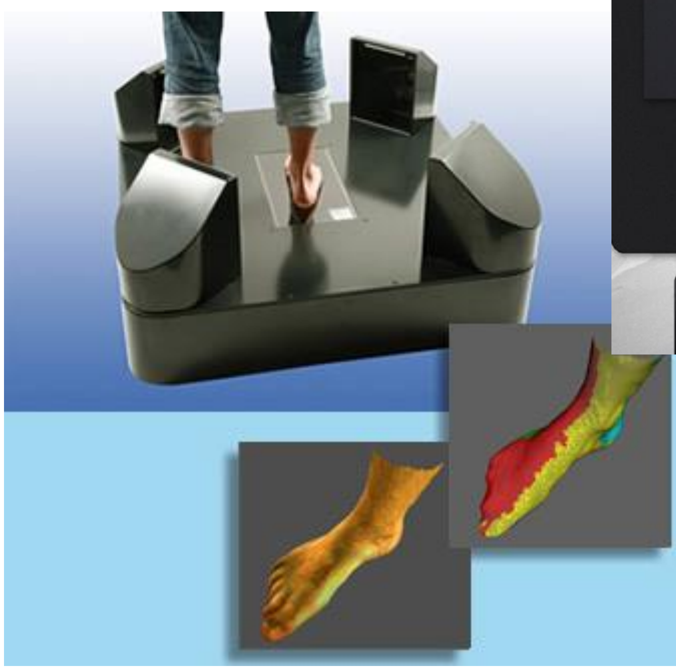


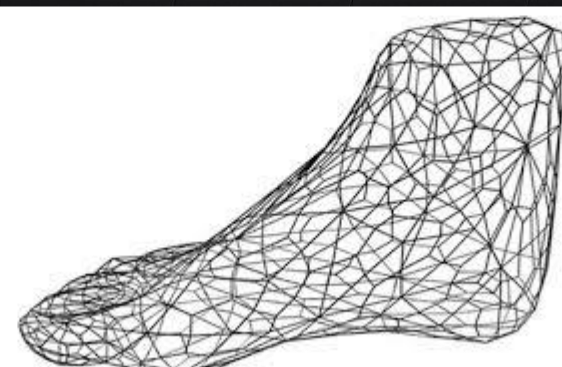
Source: Converting 3D Furniture Models to Fabricable Parts and Connectors, Lau et al., Siggraph 2011

Methods for Designing Reduced Design Spaces

- Parametric modeling
- Procedural modeling
- Deformation methods

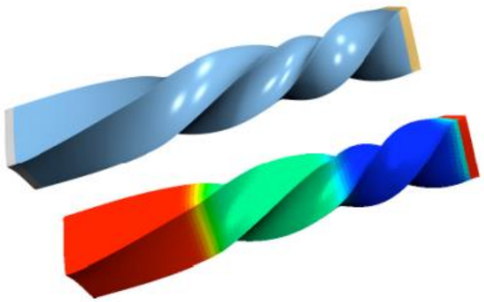
CAD Does Not Work For Organic Shapes





Deformation: Common Paradigms

- Surface based deformation
  - Optimization on the surface
  - Physically motivated: variants of elastic energy minimization



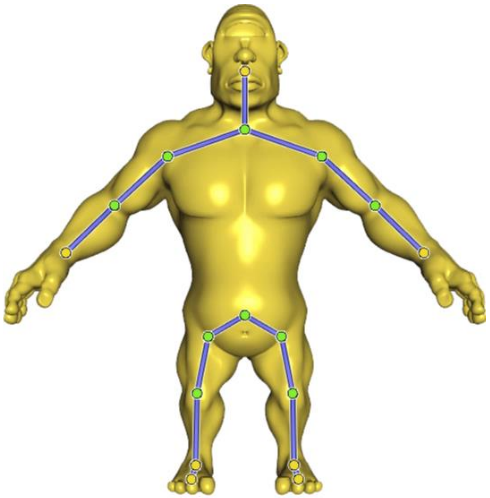
- Space deformation
  - Deforms some 2D/3D space using a cage
  - Deformation propagation to all points in the space
  - Independent of shape representation

Source: Olga Sorkine-Hornung, Daniele Panozzo

Geometric Deformation Models

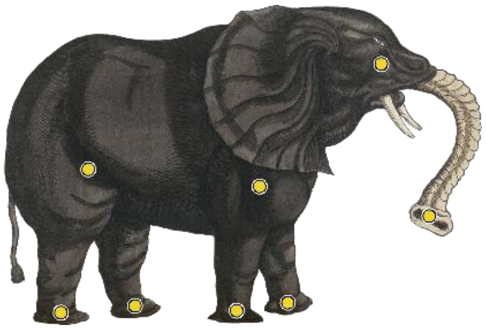
- Surface-Based Deformations
  - Laplacian Mesh Deformation
  - As Rigid As Possible Deformation (ARAP)
- Space-Based Deformations
  - Cages
  - Linear Blend Skinning (LBS)
  - Bounded Biharmonic Weights

Reduced Parameters

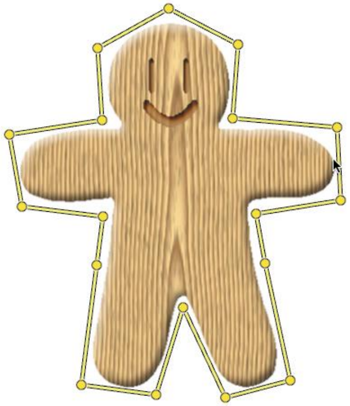


skeletons

regions

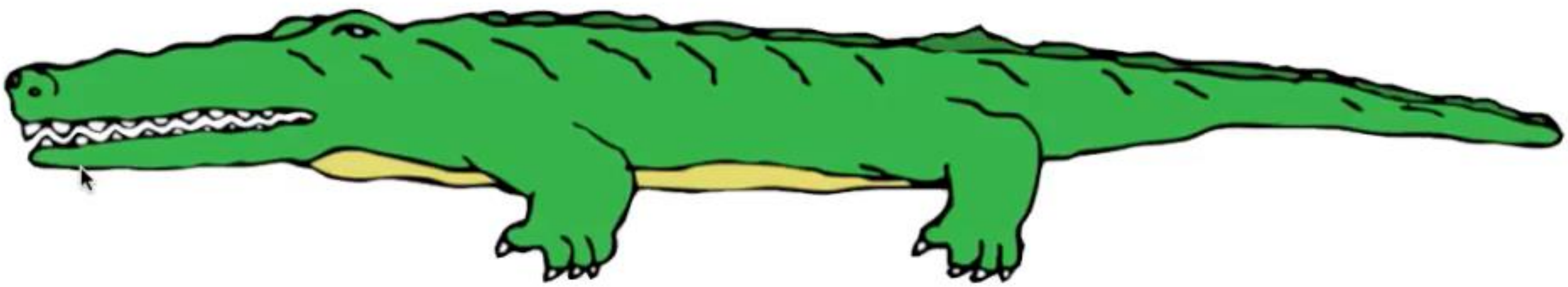


points

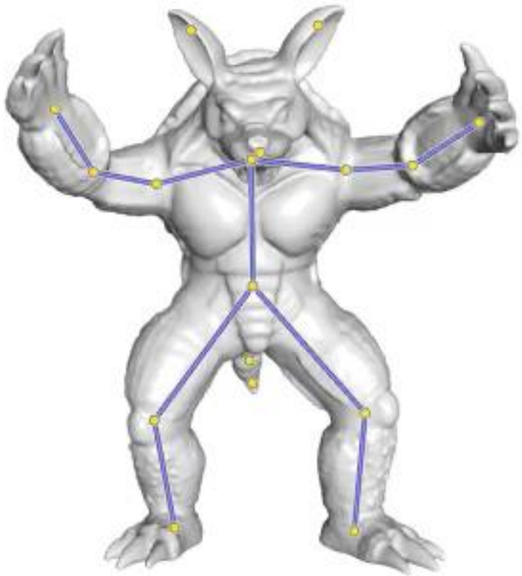


cages

Bounded Biharmonic Weights



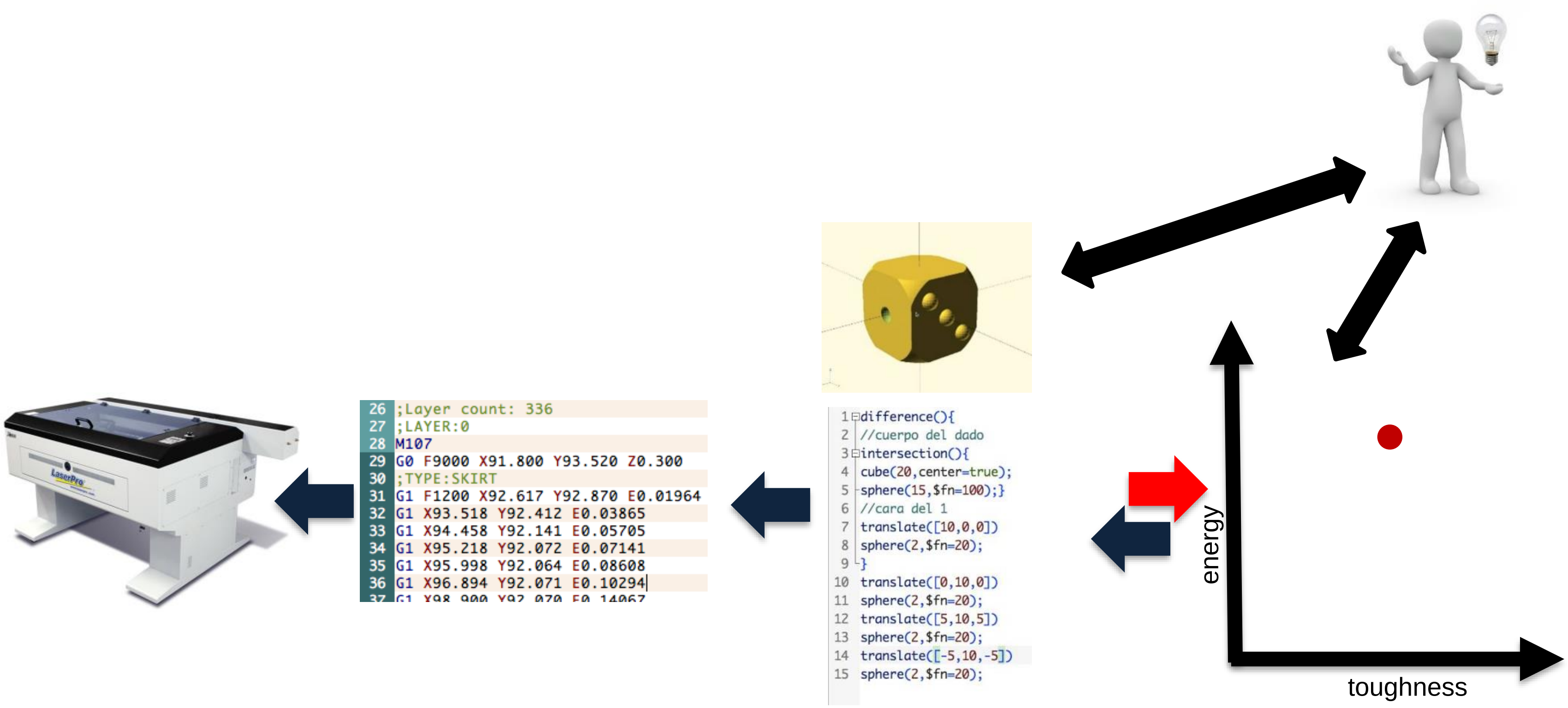
3D Characters



Course Schedule

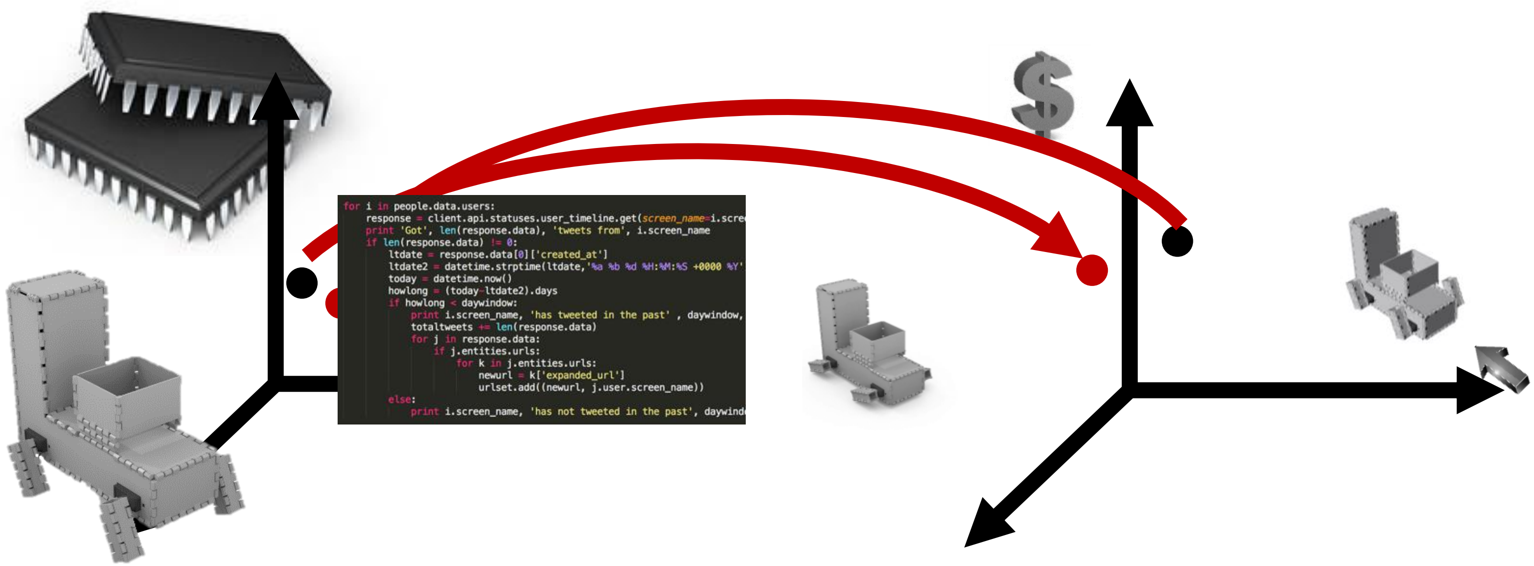
9:00 am – 9:10 am	Introduction
9:10 am – 9:25 am	Hardware Review
9:25 am – 9:50 am	From Design to Machine Code
9:50 am – 10:20 am	Design Space Representations
10:20 am – 10:25 am	Performance Driven Design
10:25 am - 10:40 am	Break
10:40 am – 10:55 am	Performance Space Representation
10:55 am - 11:15 am	Inverse Methods
11:15 am - 11:35 am	Multi-objective Inverse Methods
11:35 am - 12:00 pm	Advanced Performance Driven Design
12:00 pm - 12:15 pm	Course Review

Computational Design Stack



Hardware **Machine Code** **Design** **Performance**

Design Driven By Performance

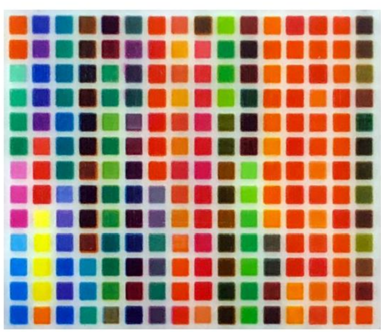


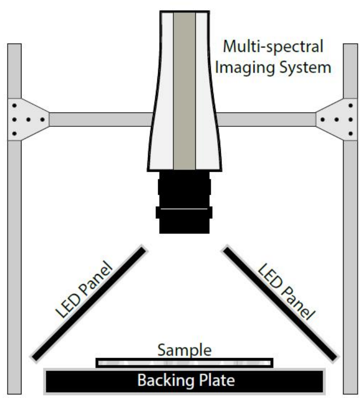
Design Space

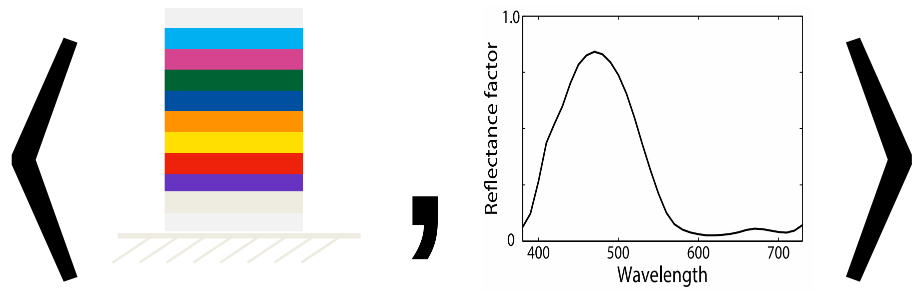
Performance Space

From Design to Performance

- Manufacturing + Testing





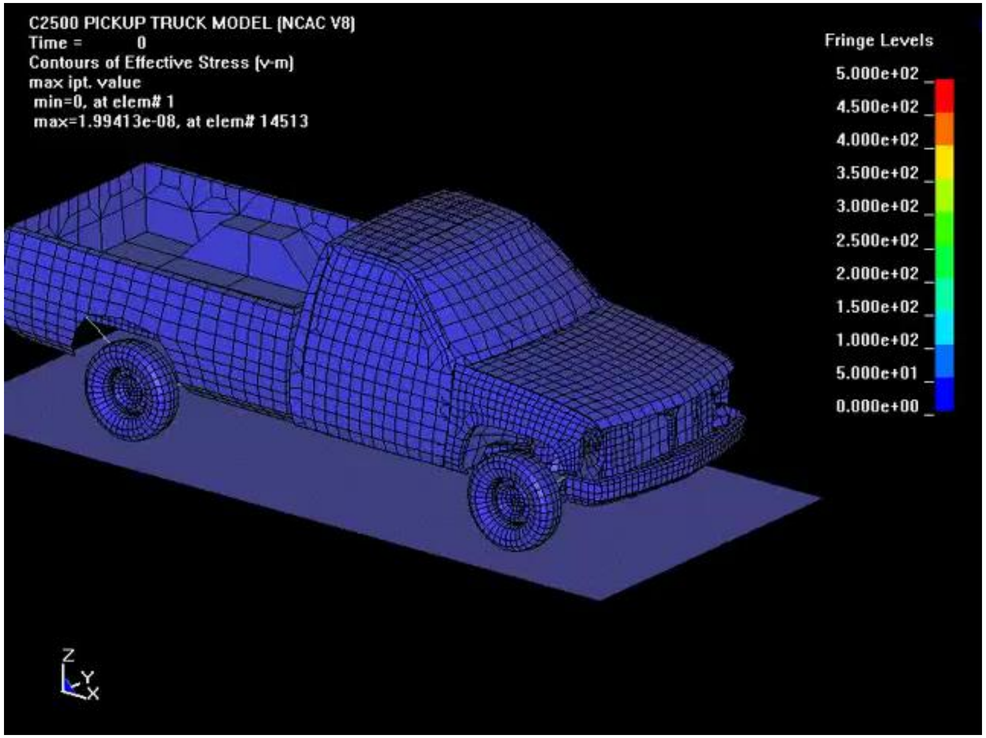


Layout Design

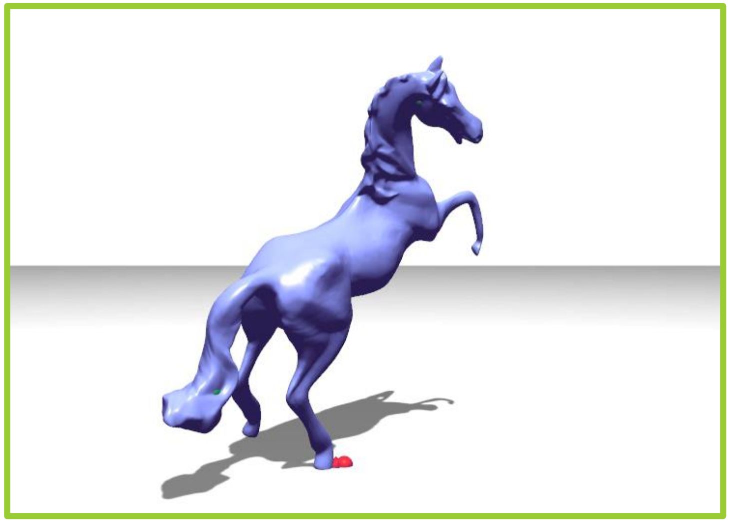
Spectrum Performance

Many Different Simulation Types & Methods

- Mechanical
  - dynamic
  - static
- Acoustic
- Thermal
- Electromagnetic
- etc.



Simulating Mechanical Behavior: Structural Stability



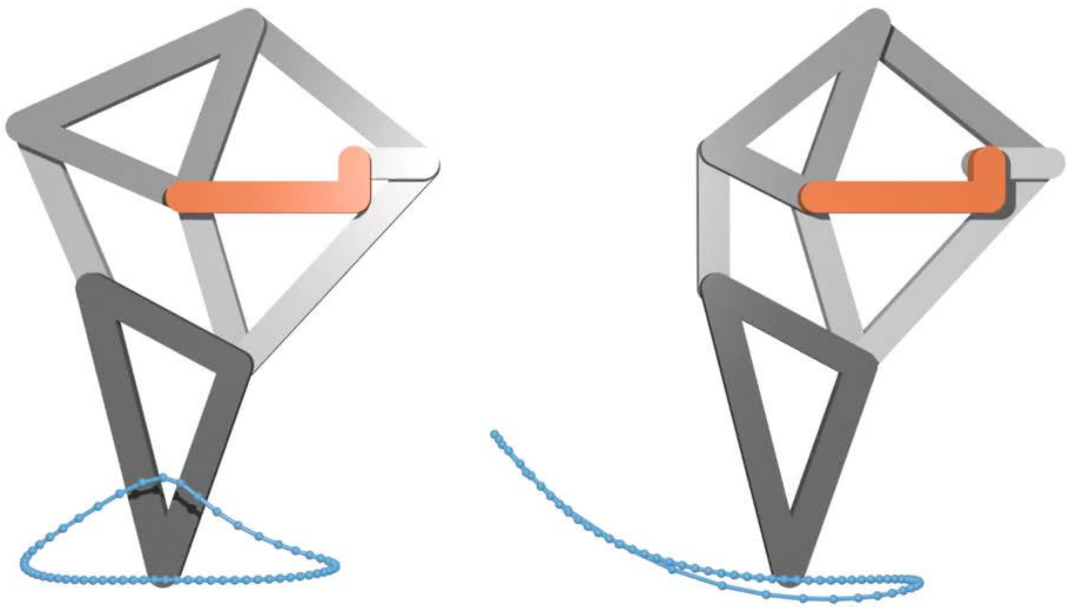
Source: Prevost et al 2013

Simulating Mechanical Behavior

Worse-Case Structural Analysis

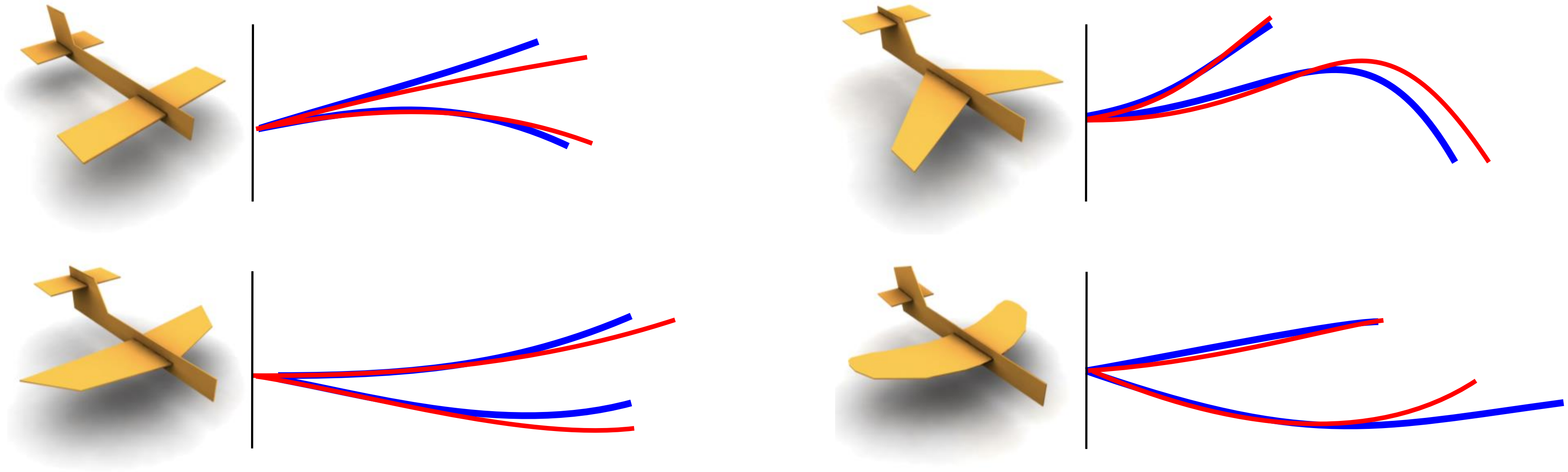
Zhou et al 2013

Simulating Kinematics



Bacher et al 2014

Simulating Aerodynamics



Umetani et al 2014

Course Schedule

9:00 am – 9:10 am	Introduction
9:10 am – 9:25 am	Hardware Review
9:25 am – 9:50 am	From Design to Machine Code
9:50 am – 10:20 am	Design Space Representations
10:20 am – 10:25 am	Performance Driven Design
10:25 am - 10:40 am	Break
10:40 am – 10:55 am	Performance Space Representation
10:55 am - 11:15 am	Inverse Methods
11:15 am - 11:35 am	Multi-objective Inverse Methods
11:35 am - 12:00 pm	Advanced Performance Driven Design
12:00 pm - 12:15 pm	Course Review