

# PL Techniques for 3D Printing



Chandakana  
Nandi



Anat  
Caspi



Dan  
Grossman



Zachary  
Tatlock

Compilers generate  
our environment.

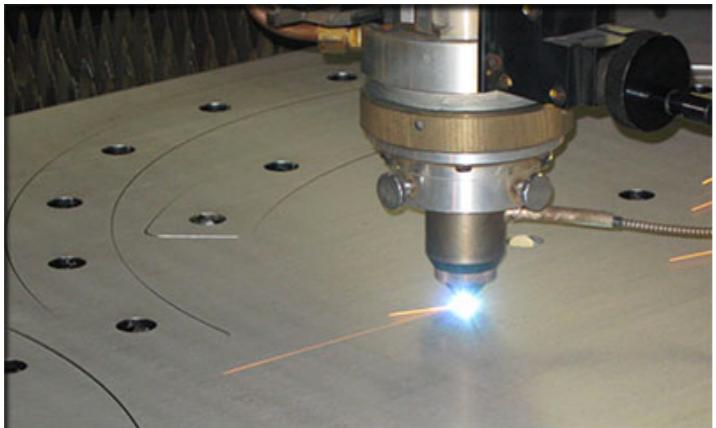






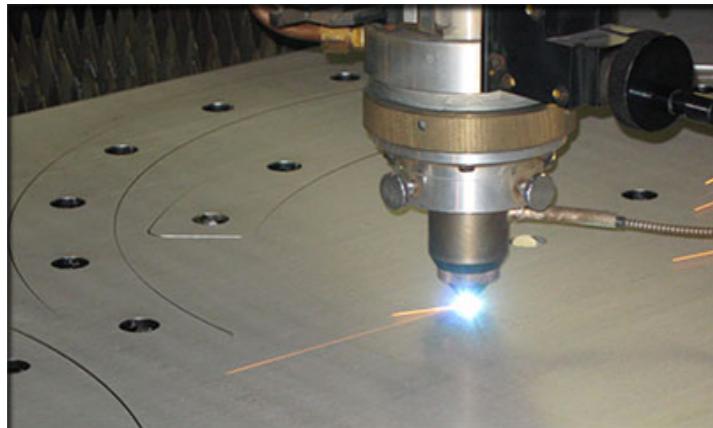


# Computerized Manufacturing



CNC

# Computerized Manufacturing



CNC

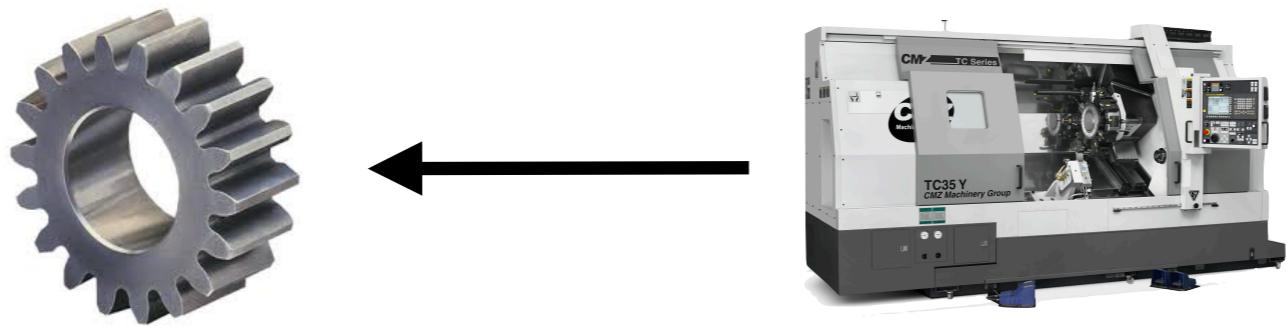


Mold Making

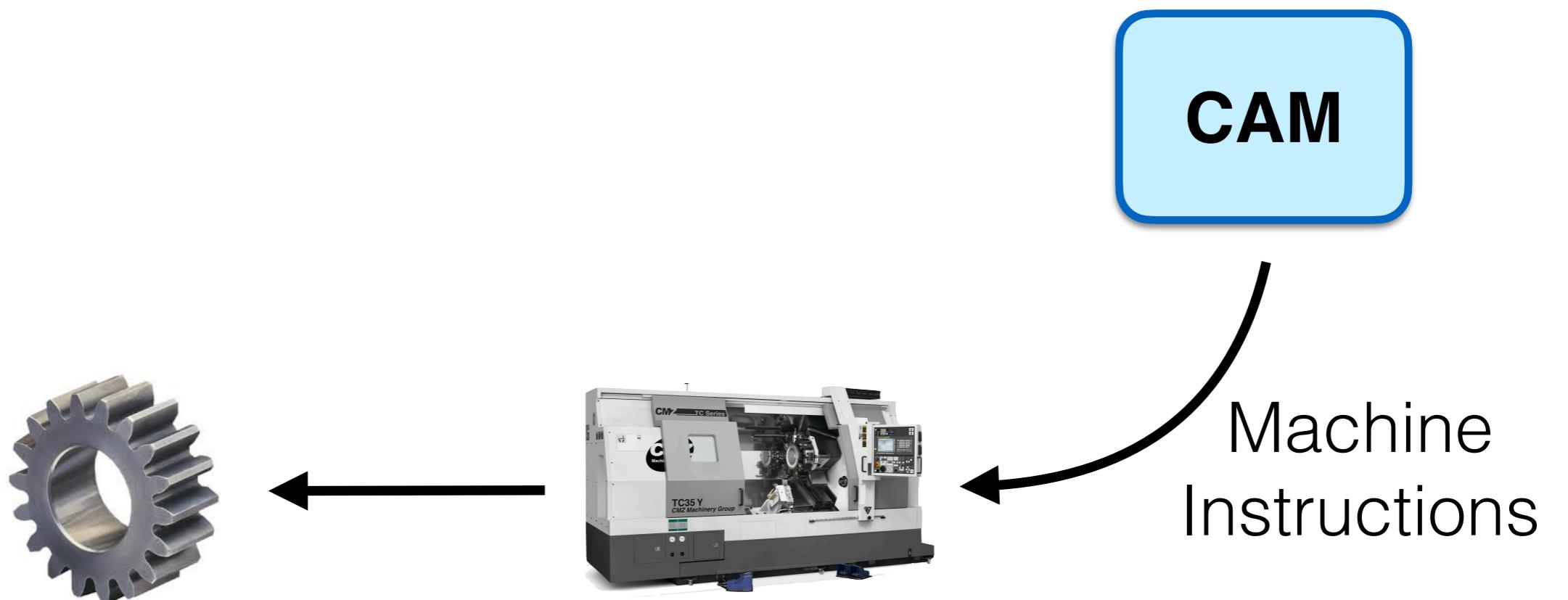


Robotic Assembly

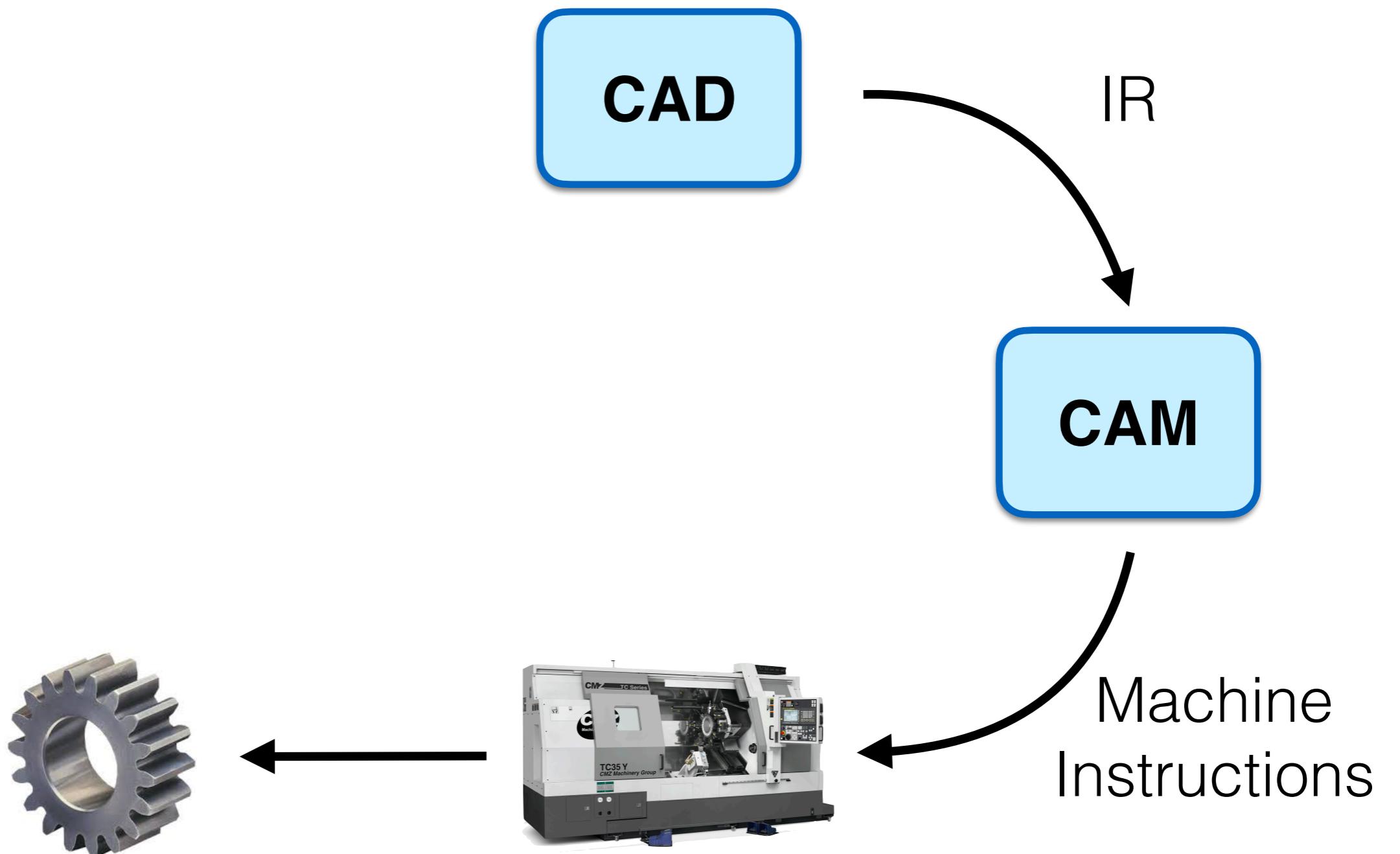
# Computerized Manufacturing



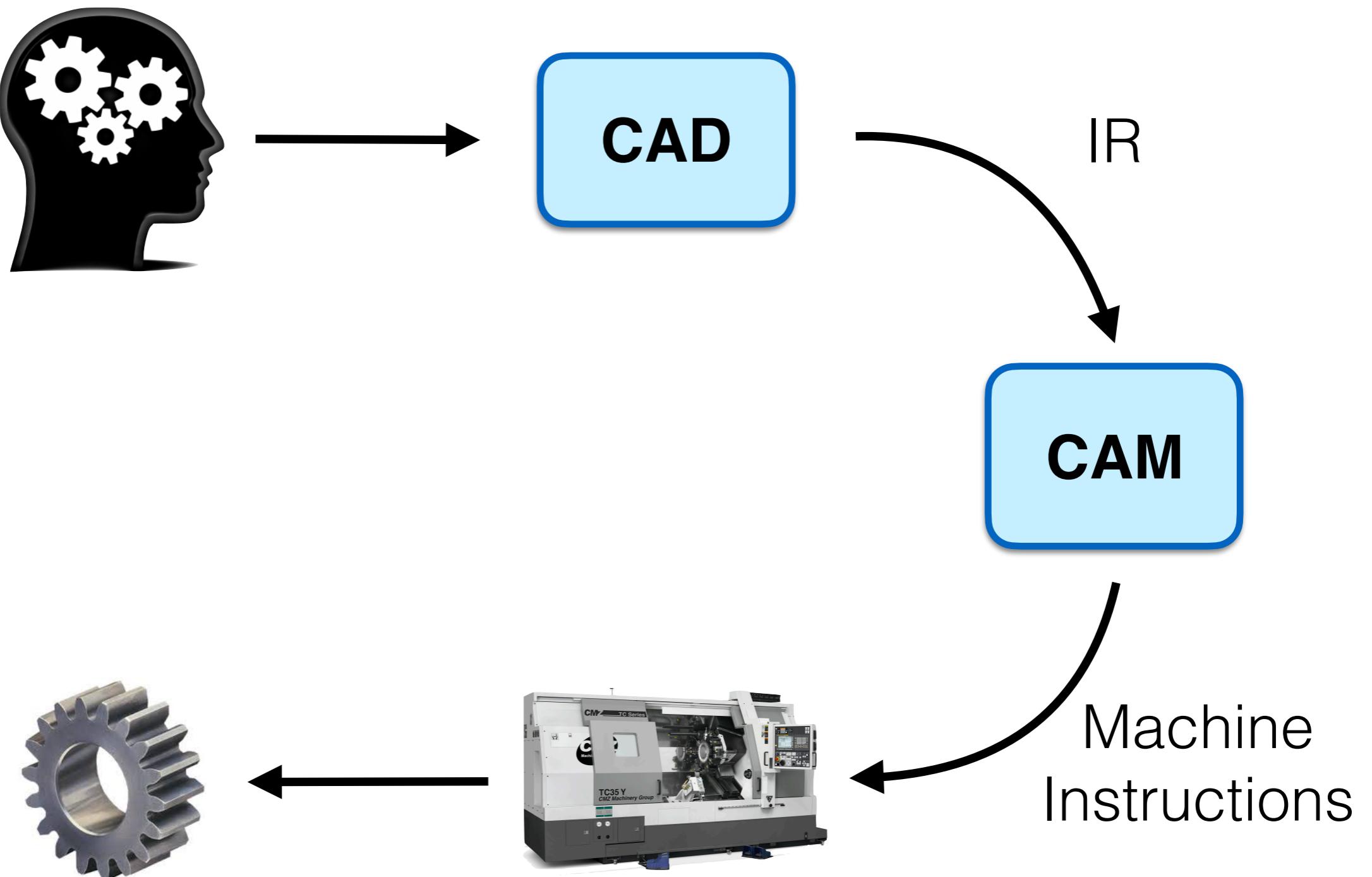
# Computerized Manufacturing



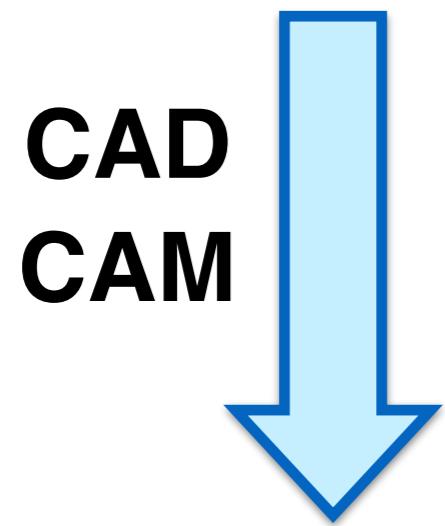
# Computerized Manufacturing



# Computerized Manufacturing



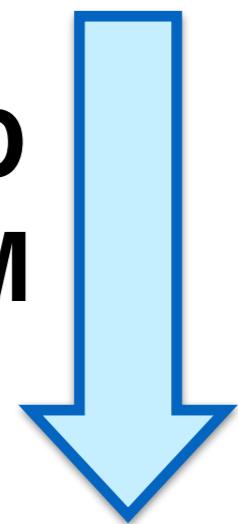
# CAD/CAM : Idea → Part



# CAD/CAM : Idea → Part



**CAD  
CAM**



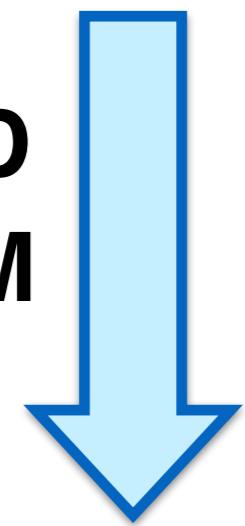
Where is the PL theory?

- semantics
- equivalence
- refinement
- approximation

# CAD/CAM : Idea → Part



**CAD  
CAM**

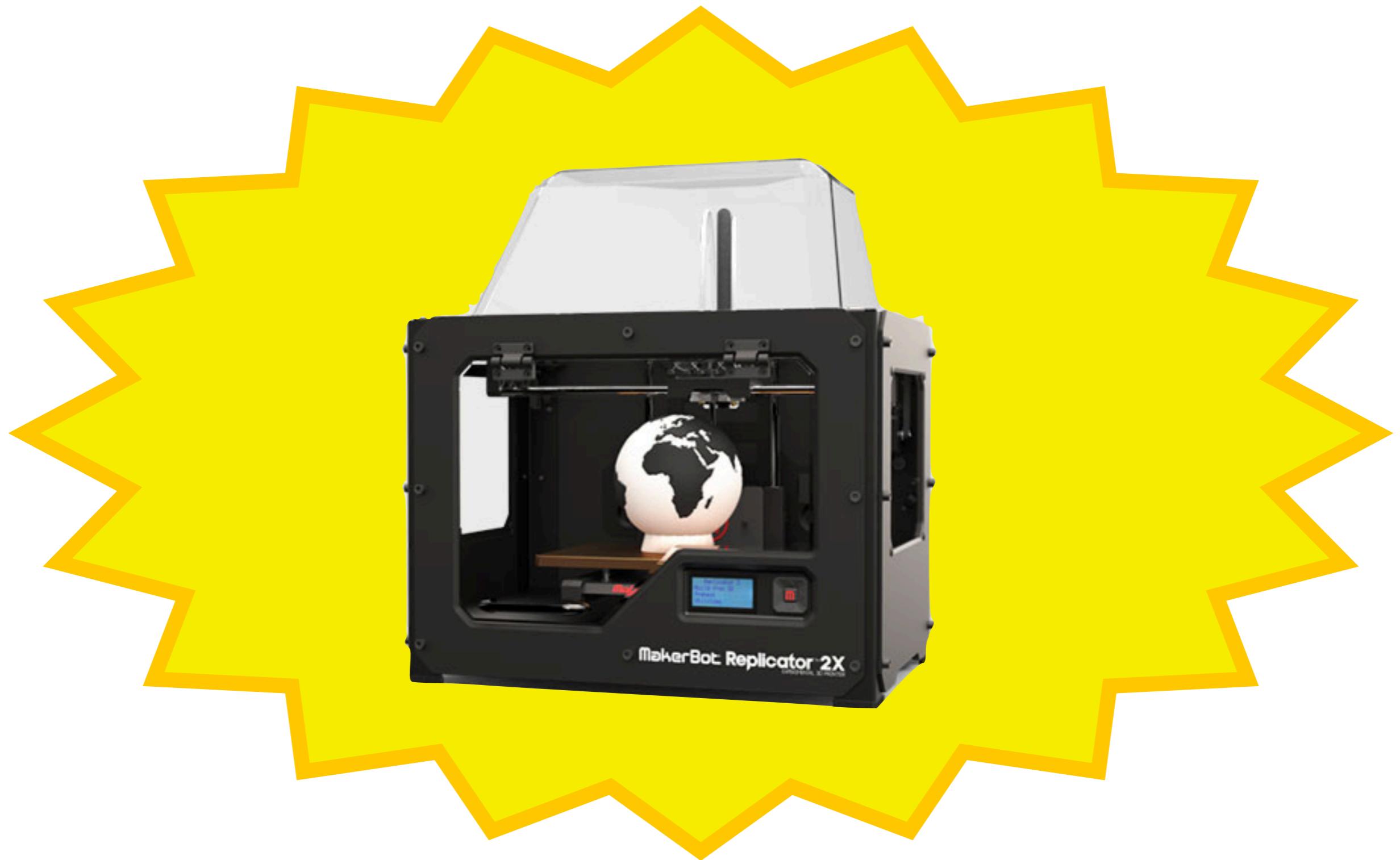


Where is the PL theory?

- semantics
- equivalence
- refinement
- approximation

Already worthy challenge,  
but recently...

# Democratized Manufacturing

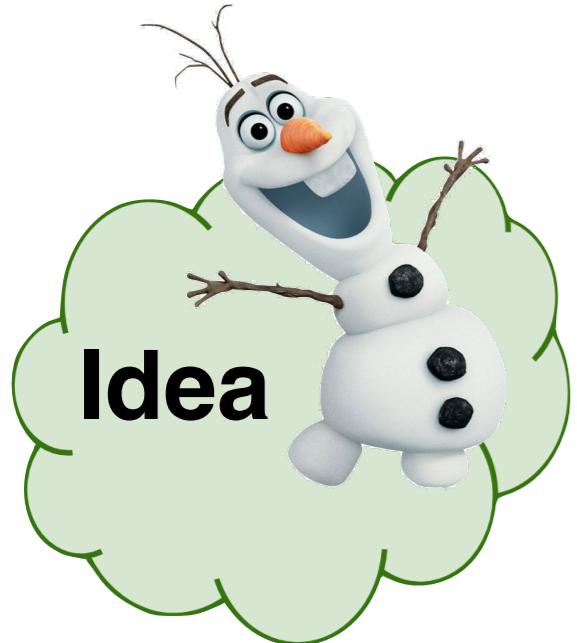


# 3DP: PL Opportunity

3D Printing Background

Challenge: CAD Synthesis

Challenge: Slicing Framework

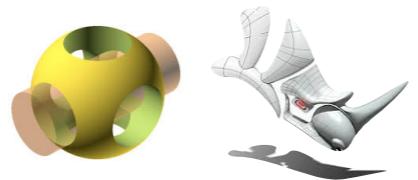


# 3D Printing Workflow





# 1. Design

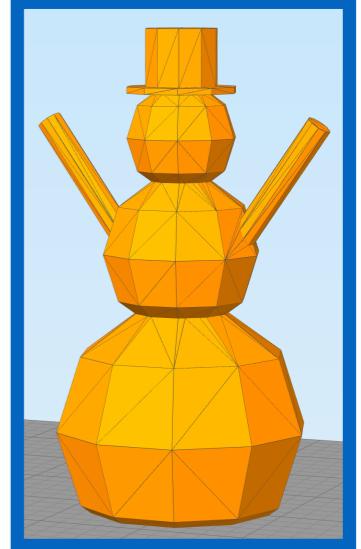


CAD

```
module snowman(scale, armAng) {  
    rs = [scale, scale / 1.6, scale / 2.3];  
    chopBase(0.65 * rs[0]) {  
        sphere(r = rs[0]);  
        translate([0, 0, 0.85 * (rs[0] +  
            sphere(r = rs[1]));  
        translate([0, 0, 0.85 * (rs[1] +  
            sphere(r = rs[2]));  
        translate([0, 0, 0.8 * rs[2]]);  
        hat(scale);  
    };  
    scale, armAng);  
};  
or()  
arm(scale, armAng);
```

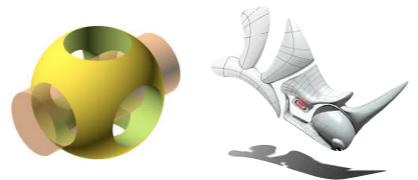


STL





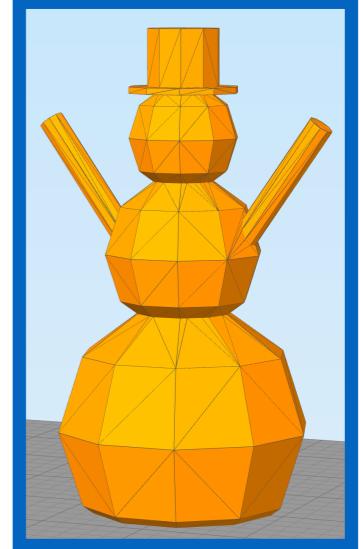
# 1. Design



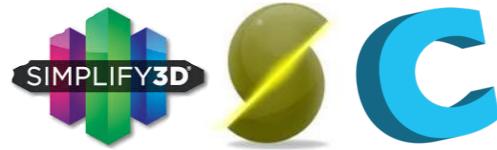
CAD

```
module snowman(scale, armAng) {  
    rs = [scale, scale / 1.6, scale / 2.3];  
    chopBase(0.65 * rs[0]) {  
        sphere(r = rs[0]);  
        translate([0, 0, 0.85 * (rs[0] +  
            sphere(r = rs[1]));  
        translate([0, 0, 0.85 * (rs[1] +  
            sphere(r = rs[2]));  
        translate([0, 0, 0.8 * rs[2]]);  
        hat(scale);  
    };  
    scale, armAng);  
};  
or()  
arm(scale, armAng);
```

STL



# 2. Slice



```
G1 X97.097 Y100.000 F6000  
G1 E0.0000 F2400  
G92 E0  
G1 X97.239 Y99.103 E0.0136 F412  
G1 X97.651 Y98.294 E0.0272  
G1 X98.294 Y97.651 E0.0408  
G1 X99.103 Y97.239 E0.0544  
G1 X100.000 Y97.097 E0.0680  
G1 X100.897 Y97.239 E0.0816  
G1 X101.706 Y97.651 E0.0952  
G1 X102.349 Y98.294 E0.1088  
G1 X102.761 Y99.103 E0.1223  
G1 X102.903 Y100.000 E0.1359  
G1 X102.761 Y100.897 E0.1495  
G1 X102.349 Y101.706 E0.1631
```

G-code



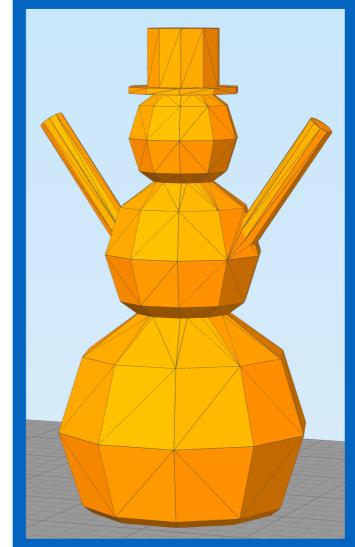
## 1. Design



CAD

```
module snowman(scale, armAng) {  
    rs = [scale, scale / 1.6, scale / 2.3];  
    chopBase(0.65 * rs[0]) {  
        sphere(r = rs[0]);  
        translate([0, 0, 0.85 * (rs[0] +  
            sphere(r = rs[1]));  
        translate([0, 0, 0.85 * (rs[1] +  
            sphere(r = rs[2]));  
        translate([0, 0, 0.8 * rs[2]]);  
        hat(scale);  
    };  
    scale, armAng);  
    or() arm(scale, armAng);  
};
```

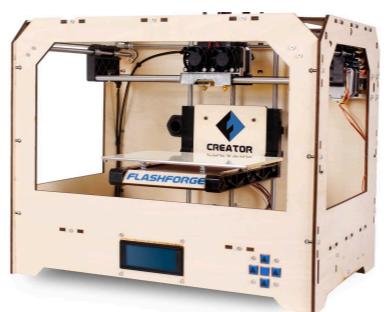
STL



## 2. Slice



## 3. Print



Marlin

SAILFISH

Part

```
G1 X97.097 Y100.000 F6000  
G1 E0.0000 F2400  
G92 E0  
G1 X97.239 Y99.103 E0.0136 F412  
G1 X97.651 Y98.294 E0.0272  
G1 X98.294 Y97.651 E0.0408  
G1 X99.103 Y97.239 E0.0544  
G1 X100.000 Y97.097 E0.0680  
G1 X100.897 Y97.239 E0.0816  
G1 X101.706 Y97.651 E0.0952  
G1 X102.349 Y98.294 E0.1088  
G1 X102.761 Y99.103 E0.1223  
G1 X102.903 Y100.000 E0.1359  
G1 X102.761 Y100.897 E0.1495  
G1 X102.349 Y101.706 E0.1631
```

G-code



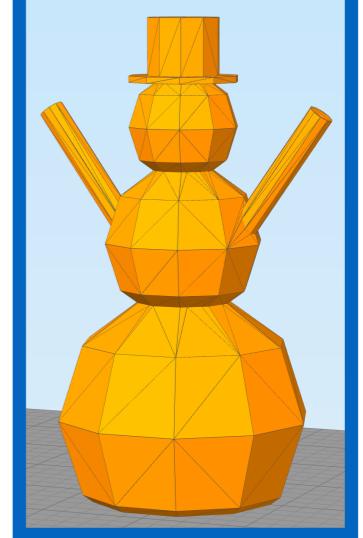
# 1. Design



CAD

```
module snowman(scale, armAng) {  
    rs = [scale, scale / 1.6, scale / 2.3];  
    chopBase(0.65 * rs[0]) {  
        sphere(r = rs[0]);  
        translate([0, 0, 0.85 * (rs[0] +  
            sphere(r = rs[1]));  
        translate([0, 0, 0.85 * (rs[1] +  
            sphere(r = rs[2]));  
        translate([0, 0, 0.8 * rs[2]]);  
        hat(scale);  
    };  
    scale, armAng);  
    arm() arm(scale, armAng);  
}
```

STL



# 4. OK?



# 3. Print



Part

# 2. Slice



```
G1 X97.097 Y100.000 F6000  
G1 E0.0000 F2400  
G92 E0  
G1 X97.239 Y99.103 E0.0136 F412  
G1 X97.651 Y98.294 E0.0272  
G1 X98.294 Y97.651 E0.0408  
G1 X99.103 Y97.239 E0.0544  
G1 X100.000 Y97.097 E0.0680  
G1 X100.897 Y97.239 E0.0816  
G1 X101.706 Y97.651 E0.0952  
G1 X102.349 Y98.294 E0.1088  
G1 X102.761 Y99.103 E0.1223  
G1 X102.903 Y100.000 E0.1359  
G1 X102.761 Y100.897 E0.1495  
G1 X102.349 Y101.706 E0.1631
```

SAILFISH

G-code



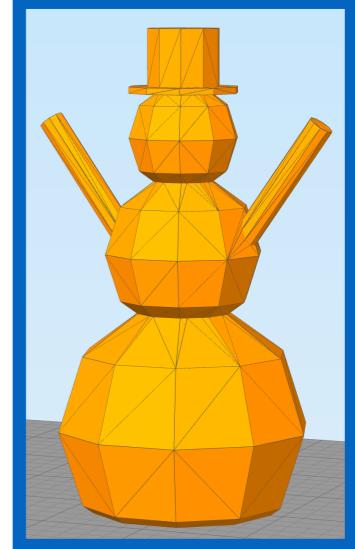
# 1. Design



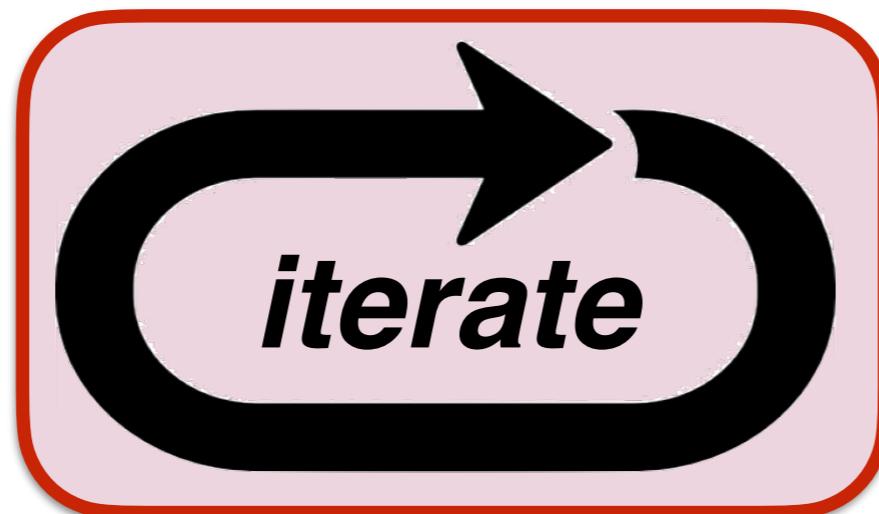
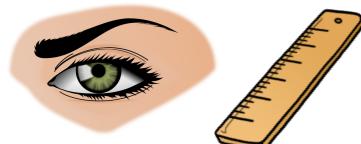
CAD

```
module snowman(scale, armAng) {
    rs = [scale, scale / 1.6, scale / 2.3];
    chopBase(0.65 * rs[0]) {
        sphere(r = rs[0]);
        translate([0, 0, 0.85 * (rs[0] +
            sphere(r = rs[1]));
        translate([0, 0, 0.85 * (rs[1] +
            sphere(r = rs[2]));
        translate([0, 0, 0.8 * rs
            hat(scale);
        ...
    }
    scale, armAng);
    or() arm(scale, armAng);
}
```

STL



# 4. OK?



# 3. Print



**Marlin**

**SAILFISH**

**Part**

# 2. Slice



```
G1 X97.097 Y100.000 F6000
G1 E0.0000 F2400
G92 E0
G1 X97.239 Y99.103 E0.0136 F412
G1 X97.651 Y98.294 E0.0272
G1 X98.294 Y97.651 E0.0408
G1 X99.103 Y97.239 E0.0544
G1 X100.000 Y97.097 E0.0680
G1 X100.897 Y97.239 E0.0816
G1 X101.706 Y97.651 E0.0952
G1 X102.349 Y98.294 E0.1088
G1 X102.761 Y99.103 E0.1223
G1 X102.903 Y100.000 E0.1359
G1 X102.761 Y100.897 E0.1495
G1 X102.349 Y101.706 E0.1631
```

**G-code**

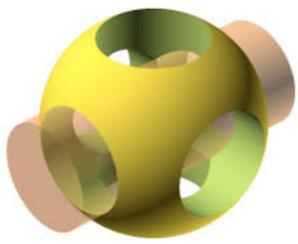
# 3DP: PL Opportunity

*3D Printing Background*

**Challenge: CAD Synthesis**

Challenge: Slicing Framework

# Challenge: CAD Synthesis



: CAD → STL



: STL → G-code

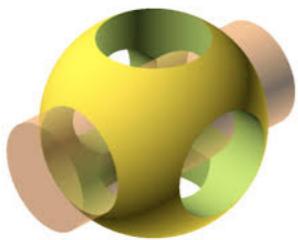


: G-code → Part

# Challenge: CAD Synthesis



: Idea → CAD



: CAD → STL



: STL → G-code



: G-code → Part

# Today: Crowdsource Designs



: Internet → Idea → STL

# Today: Crowdsource Designs



Internet → Idea → STL

A large, light blue callout shape contains two screenshots of 3D design platforms. The top screenshot is from Thingiverse, showing a "Featured" section with three lightbulbs containing portraits and a "Global Feed" with items like a "Match Head TOOTHLESS Herb GRINDER". The bottom screenshot is from GrabCAD, showing a "Recent" tab with various 3D models such as a "Legos Bricks", a "Volkswagen Golf GTI - Low P...", and a "Buggycrawler...redesign". Both screenshots show a grid of smaller thumbnail images representing different projects. To the right of the callout, the word "Easy" is written in a large, green, sans-serif font.

+ Easy

# Today: Crowdsource Designs



: Internet → Idea → **option** STL

A screenshot of the Thingiverse website is displayed within a light blue rounded rectangular frame. The website interface includes a header with navigation links like DASHBOARD, EXPLORE, EDUCATION, and CREATE, along with a search bar. Below the header are sections for Global Feed, Featured collections, and specific 3D models like a lightbulb with portraits and a 3D-printed herb grinder. The main content area shows a grid of numerous 3D models, including a Lamborghini Aventador, a CNC mill, a 36-meter sailing yacht, and various mechanical parts and toys. To the right of the screenshot, there is a large green plus sign followed by the word "Easy" in green, and below that, a large red minus sign followed by the word "Incomplete" in red.

# Today: Crowdsource Designs



: Internet → Idea → **option** STL

A screenshot of the Thingiverse website is displayed within a light blue rounded rectangular frame. The website interface includes a header with the Thingiverse logo, search bar, and navigation links (Dashboard, Explore, Education, Create). Below the header are sections for "Featured" designs (e.g., Match Head TOOTLESS Herb GRINDER) and "Global Feed" (e.g., Match Head TOOTLESS Herb GRINDER). The main content area shows a grid of various 3D models, including a lightbulb with a portrait, a small drone, a fish model, and several mechanical parts. A large green plus sign and the word "Easy" are overlaid on the top right of the frame. To the right of the frame, three red bullet points are listed: "- Incomplete", "- Hard to modify", and "- Hard to modify".

# Today: Crowdsource Designs



: Internet → Idea → **option** STL

## Goal

For idea **i**, even when:

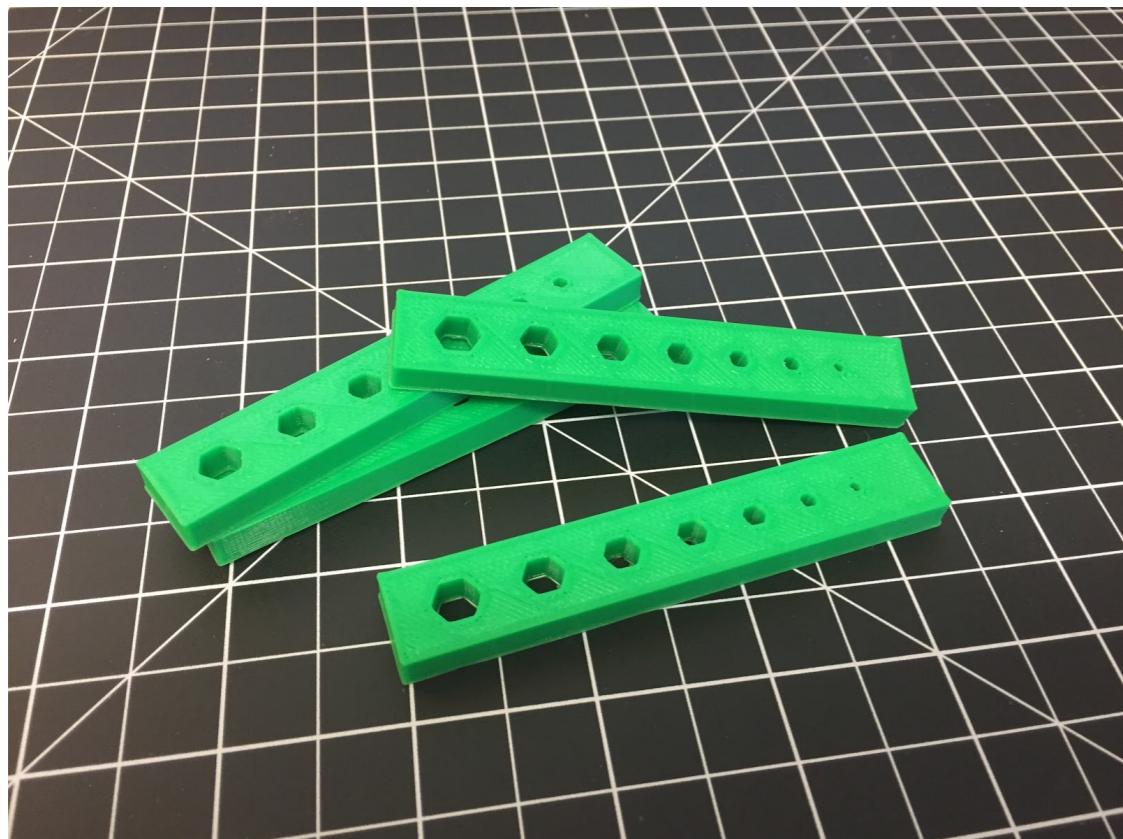
**Thingiverse** (, **i**) = **None**

there often exists similar **i'** such that:

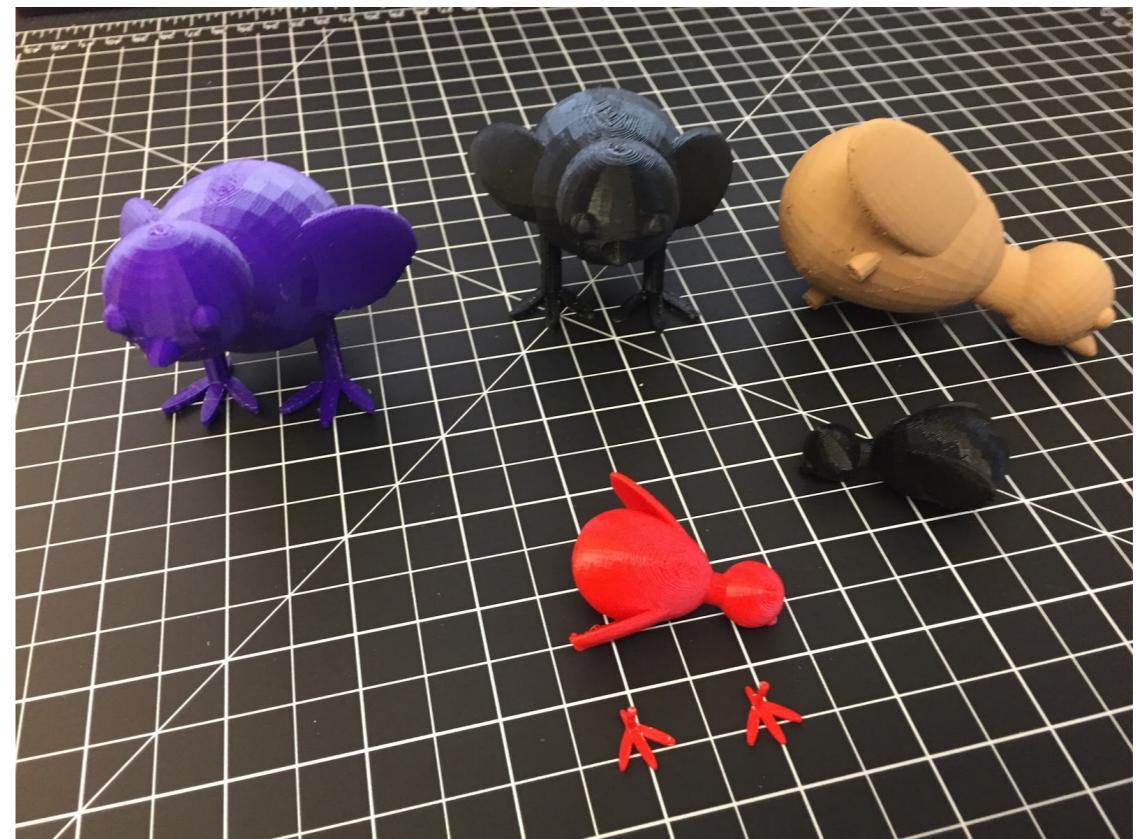
**Thingiverse** (, **i'**) = **Some s**

So: adapt “almost” design **s** to a design for **i**!

# Example “Almost” Designs

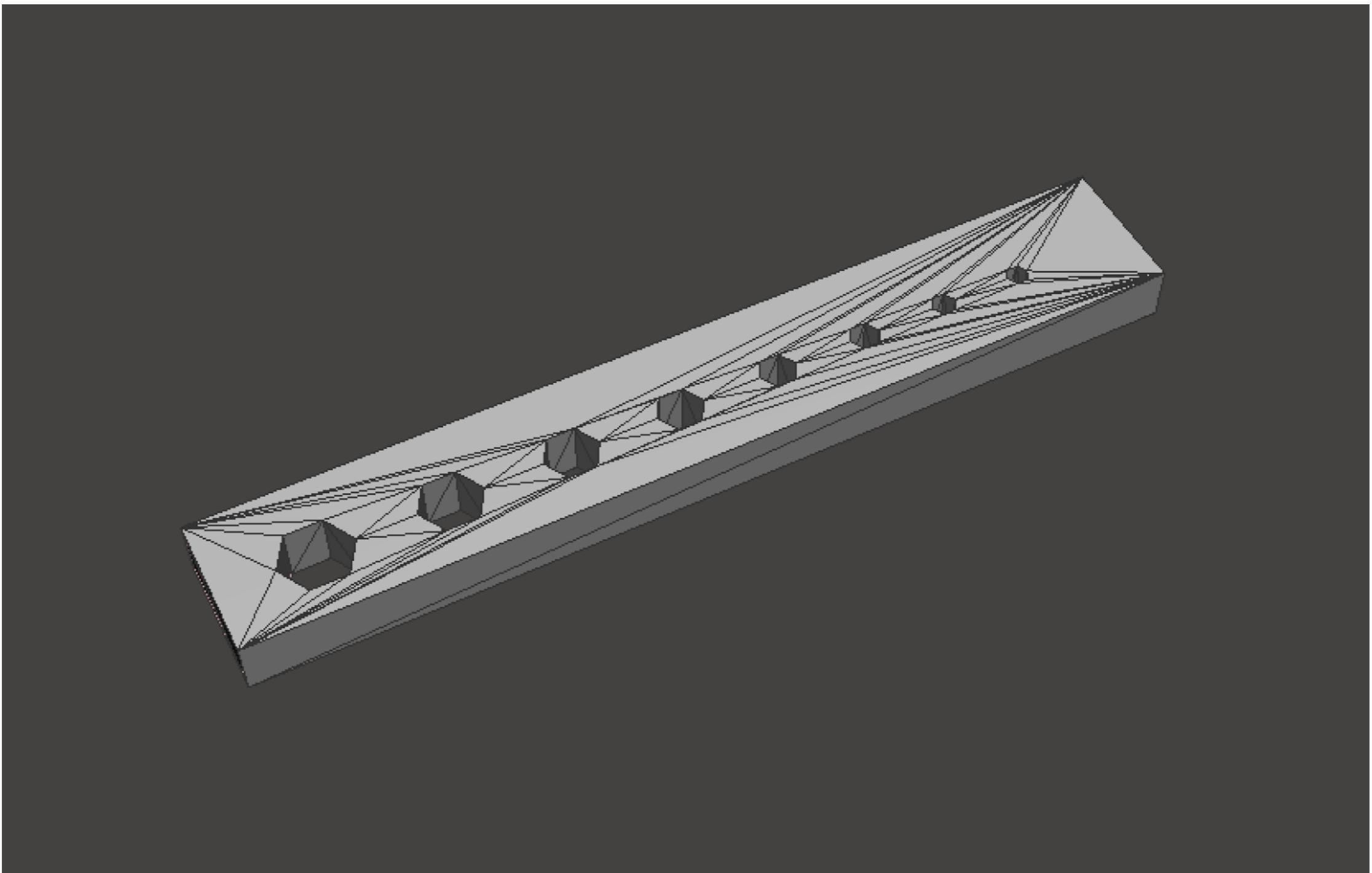


Rotated Hex Hole

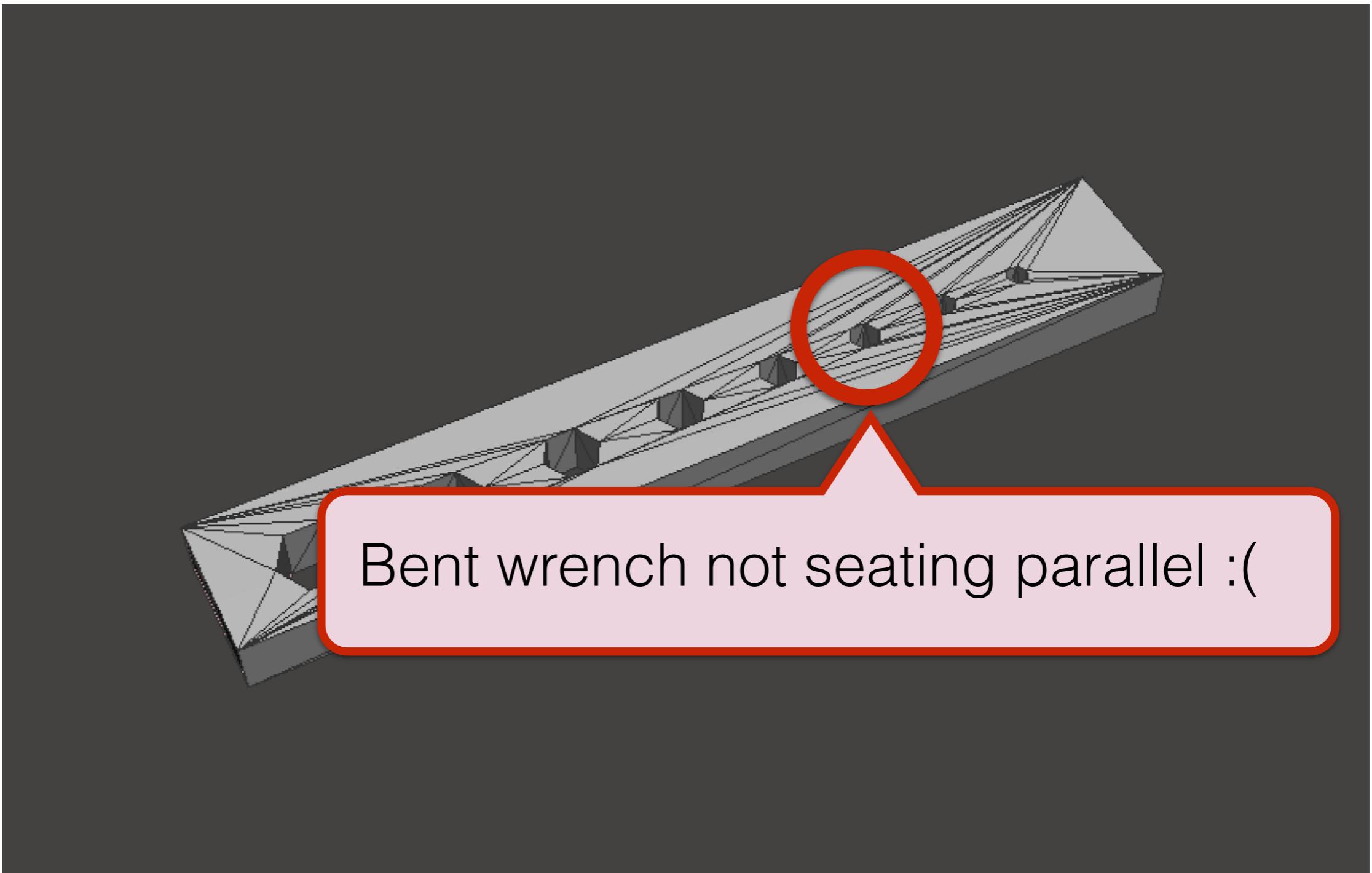


Broken Chicken Legs

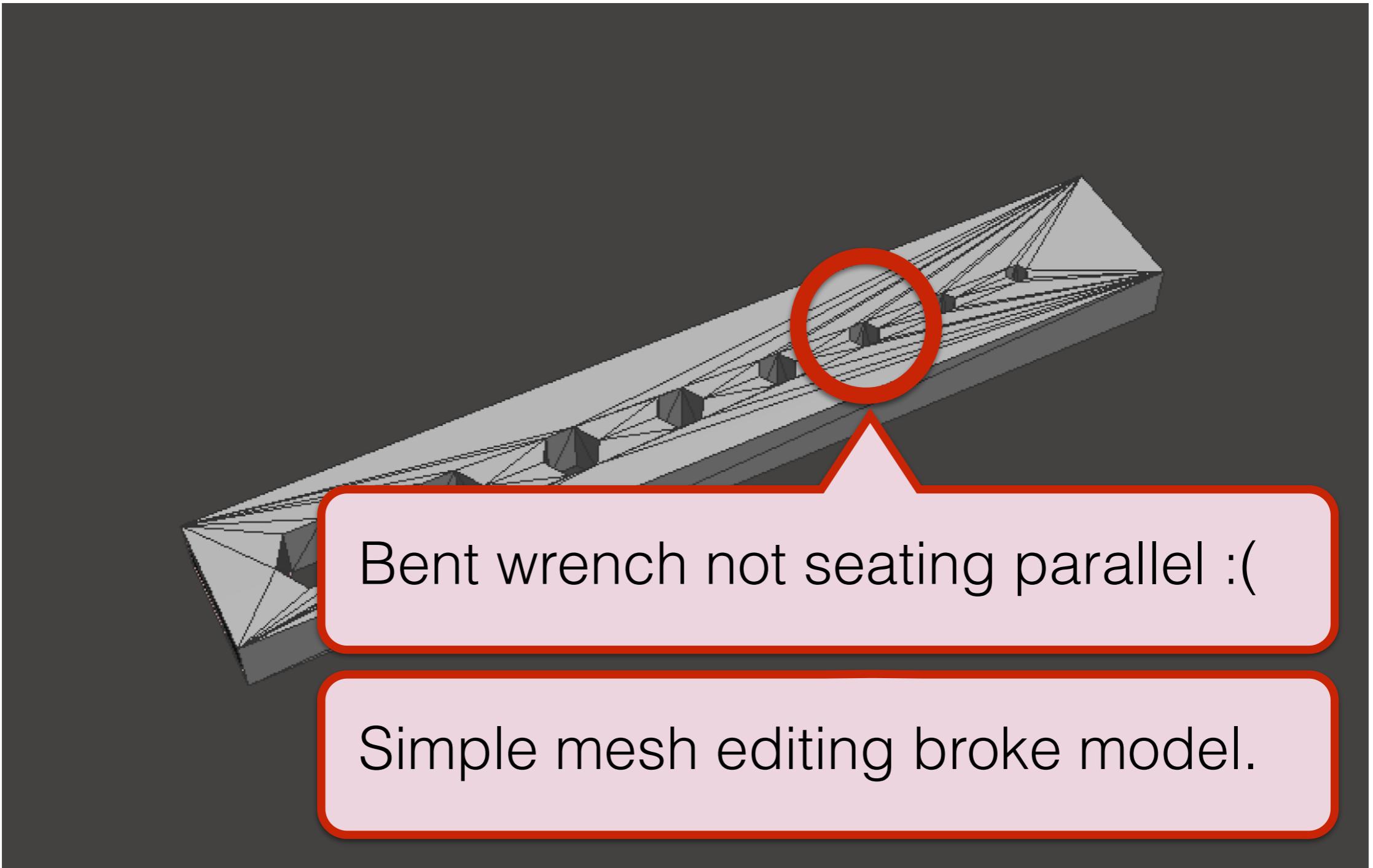
# Inferring CAD to fix hex holder



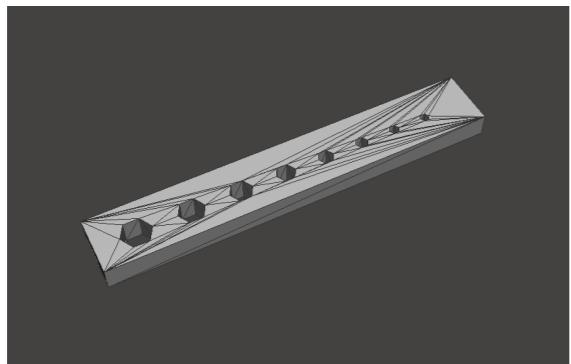
# Inferring CAD to fix hex holder



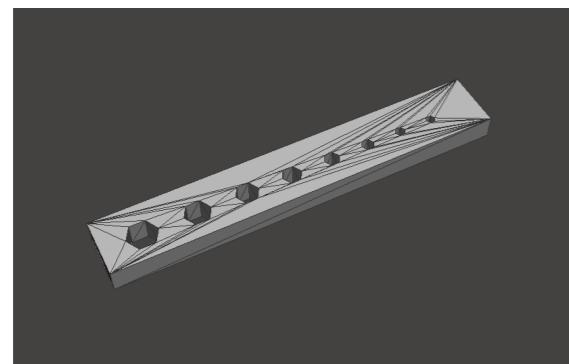
# Inferring CAD to fix hex holder



# Inferring CAD to fix hex holder



# Inferring CAD to fix hex holder

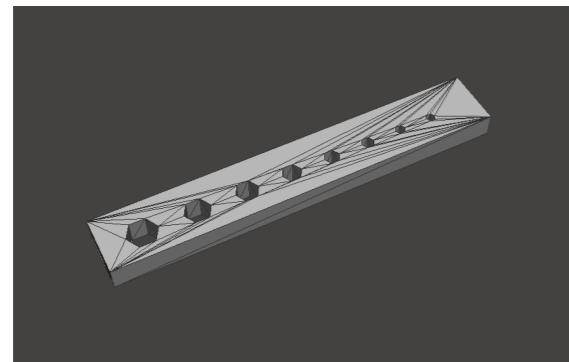


Infer  
→  
**(manual)**

```
difference() {  
    cube([w, d, h]);  
    for(i = [0 : len(holes) - 1])  
        translate([offset(i), d/2, -1])  
            hex_hole(holes[i]);  
}
```



# Inferring CAD to fix hex holder



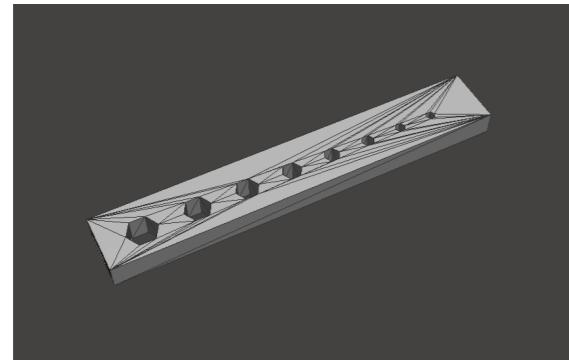
Infer  
→  
**(manual)**

```
difference() {
    cube([w, d, h]);
    for(i = [0 : len(holes) - 1])
        translate([offset(i), d/2, -1])
            hex_hole(holes[i]);
```

Tweak ↓ **(small)**

```
difference() {
    cube([w, d, h]);
    for(i = [0 : len(holes) - 1])
        translate([offset(i), d/2, -1])
            if(i == 5)
                rotate([0, 0, 35])
                    hex_hole(holes[i]);
            hex_hole(holes[i]);
```

# Inferring CAD to fix hex holder



Infer  
**(manual)**

```
difference() {
    cube([w, d, h]);
    for(i = [0 : len(holes) - 1])
        translate([offset(i), d/2, -1])
            hex_hole(holes[i]);
```

A small yellow square containing a white hexagon icon.

Print

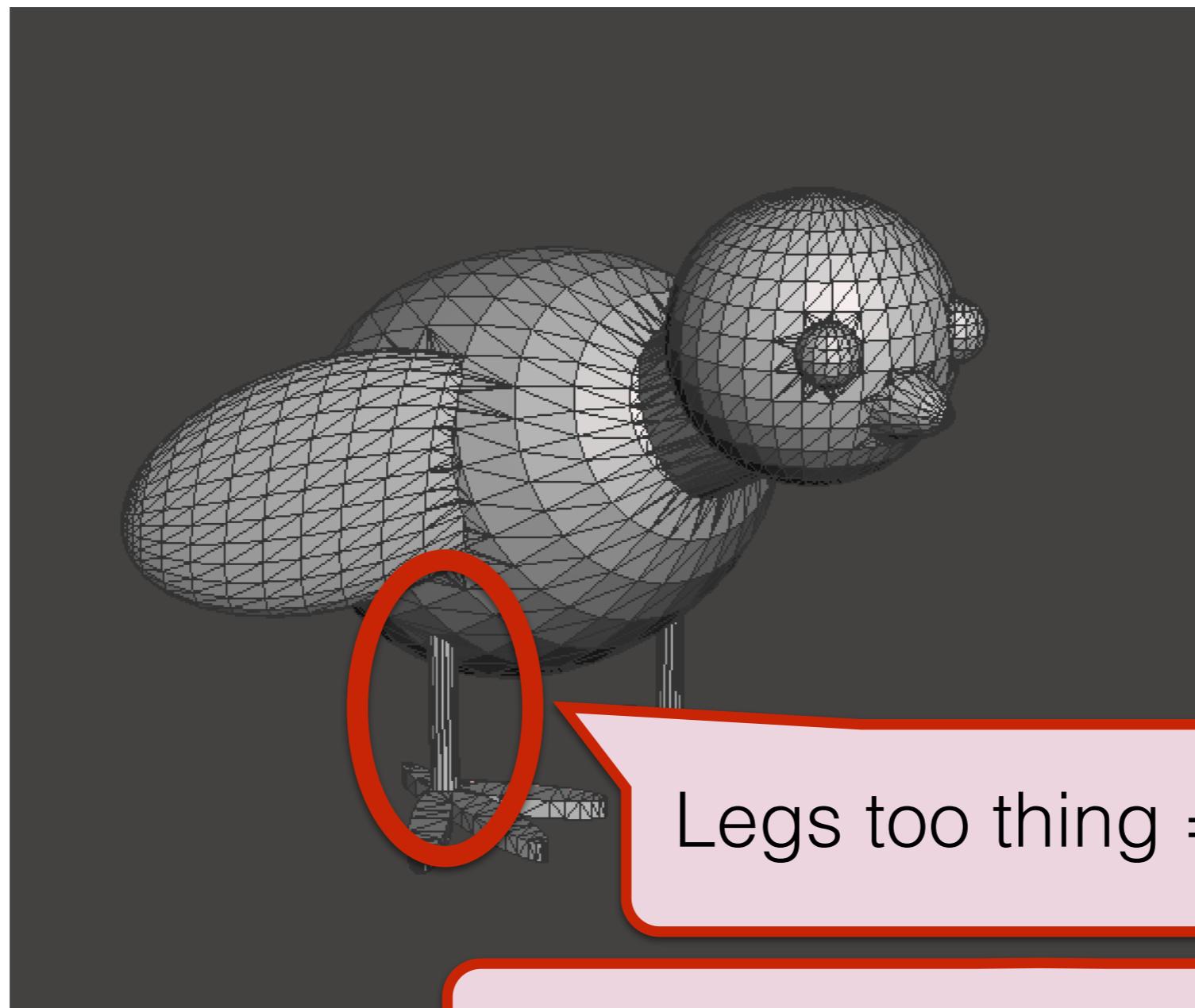
Tweak ↓ **(small)**

```
difference() {
    cube([w, d, h]);
    for(i = [0 : len(holes) - 1])
        translate([offset(i), d/2, -1])
            if(i == 5)
                rotate([0, 0, 35])
                    hex_hole(holes[i]);
            hex_hole(holes[i]);
```

A small yellow square containing a white hexagon icon.

**(success)**

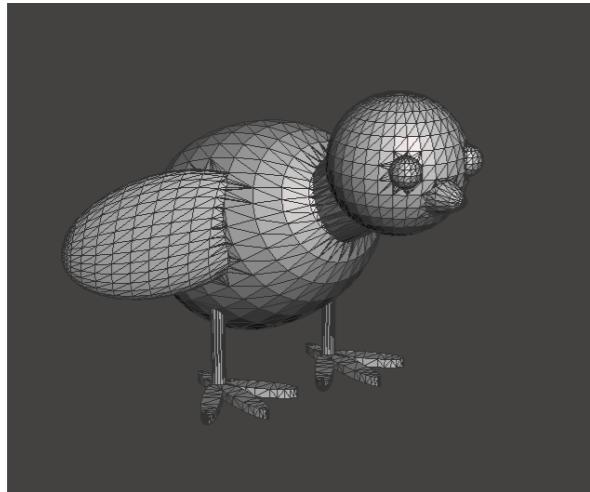
# Inferring CAD to fix chicken



Legs too thin => broke!

Expanding leg in STL tedious.

# Inferring CAD to fix chicken



Infer  
→  
**(manual)**

```
...  
cylinder(h = 30, r = 2);  
cylinder(h = 30, r = 2);  
...
```



Tweak ↓ **(small)**

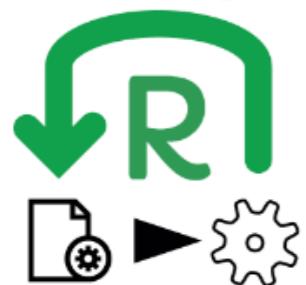
Print ←

```
...  
cylinder(h = 30, r = 4);  
cylinder(h = 30, r = 4);  
...
```

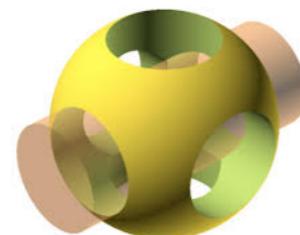
**(success)**

# Challenge: CAD Synthesis

decompiler



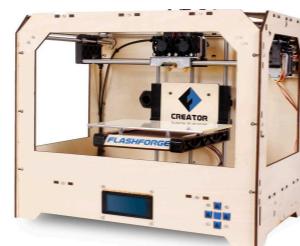
: Internet → Idea → CAD



: CAD → STL



: STL → G-code



: G-code → Part

# 3DP: PL Opportunity

*3D Printing Background*

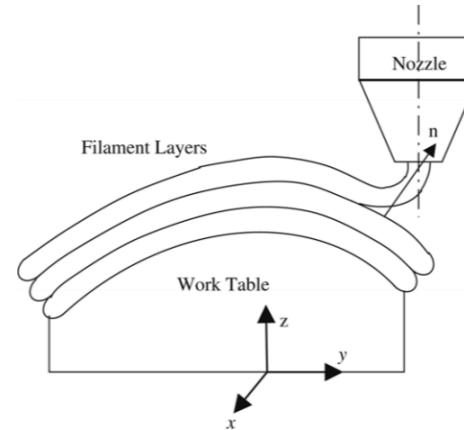
*Challenge: CAD Synthesis*

**Challenge: Slicing Framework**

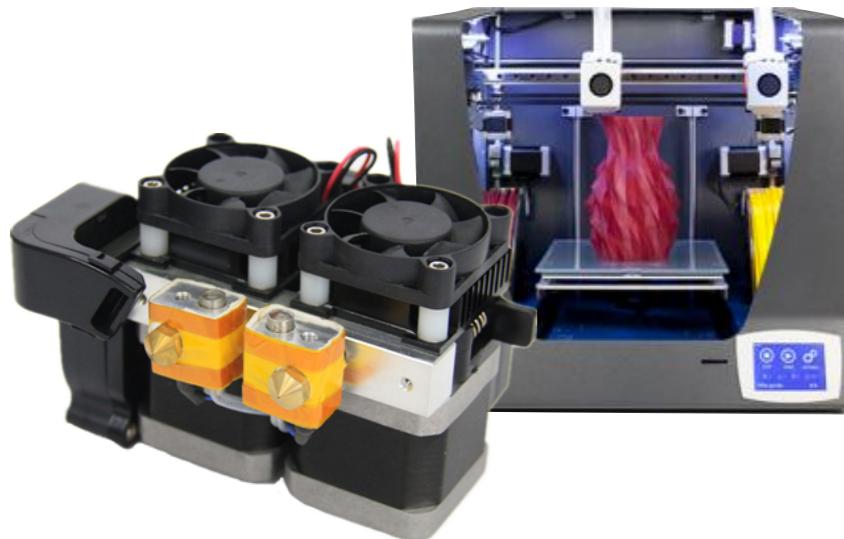
# Exploring Slicing Strategies



Partitioning [Chopper 12]



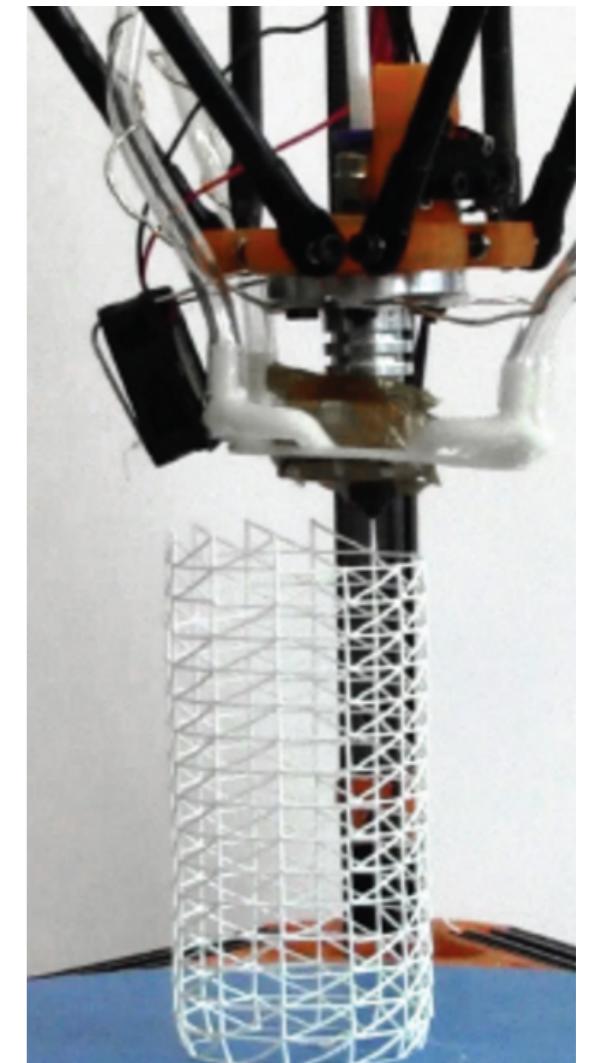
Curved Layer FDM [CAD 08]



Parallelization



Multi-material [OpenFab 13]



Approx [Wireprint 14]

# 3DP Slicing Framework

Today: roll your own framework

*Goal: LLVM / CIL for 3D Printing*

Should enable exploring many new strategies:

- error compensation
- G-code synthesis with Z3
- parallelizing peepholes
- cross-part constraint
- ...

# 3DP: PL Opportunity

Solid foundation:

- compiler theory
- fast solvers
- diverse synthesis
- num. methods

Goals:

- fab theory
- efficiency
- self-stabilize
- tools

Compilers generate  
our environment.

Compilers generate  
our environment.

PL folk can develop the  
tools to make it better.

# *Thank You!*

Compilers generate  
our environment.

PL folk can develop the  
tools to make it better.