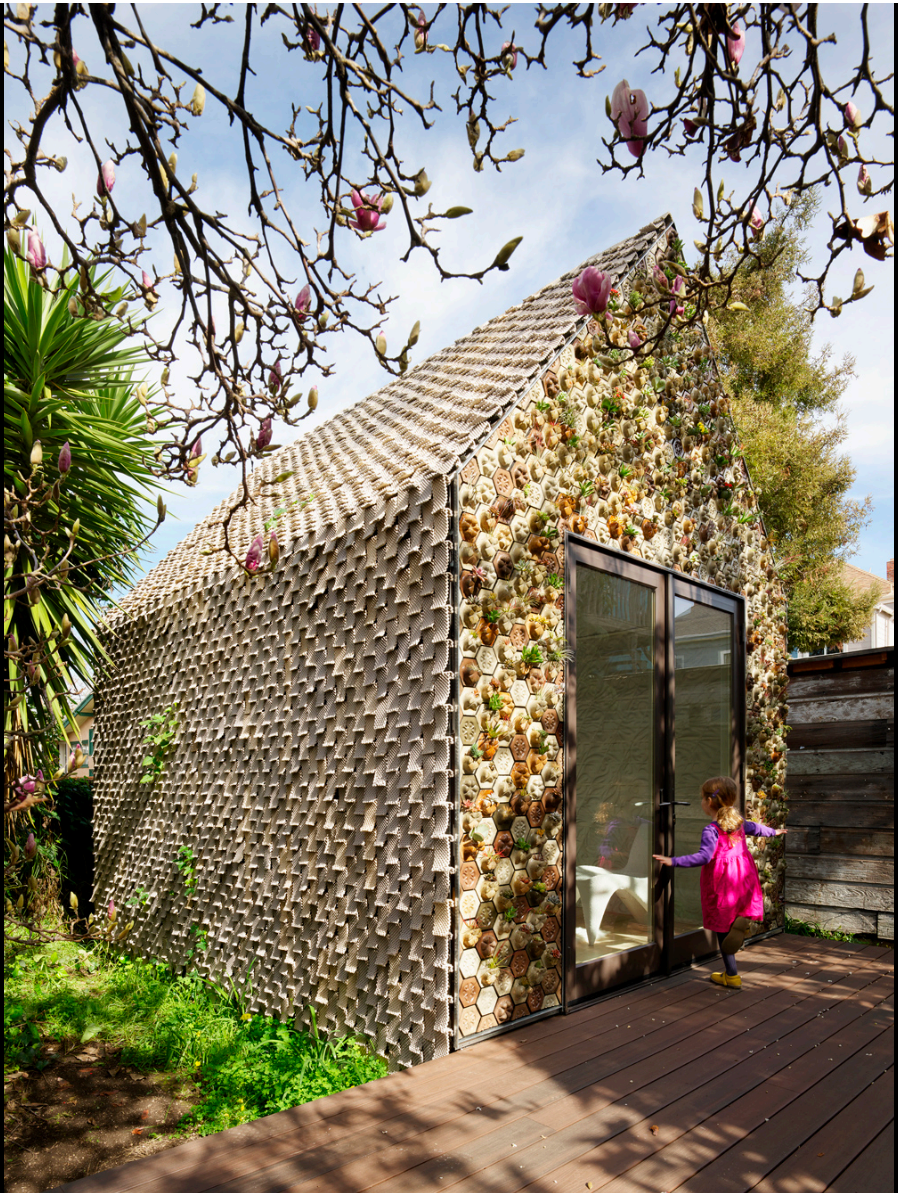
# Computational Fabrication

CS 491 and 591
Professor: Leah Buechley
https://handandmachine.cs.unm.edu/classes/Computational_Fabrication_Spring2021/

# Weekly Designers: Emerging Objects
# Virginia San Fratello + Ronald Rael

https://www.rael-sanfratello.com/

http://emergingobjects.com/



Rael San Fratello

The Cabin of 3D Printed Curiosities demonstrates that 3D printing can be beautiful, meaningful, and well crafted – not crude, fast and cheap.

Rael San Fratello

Rael San Fratello

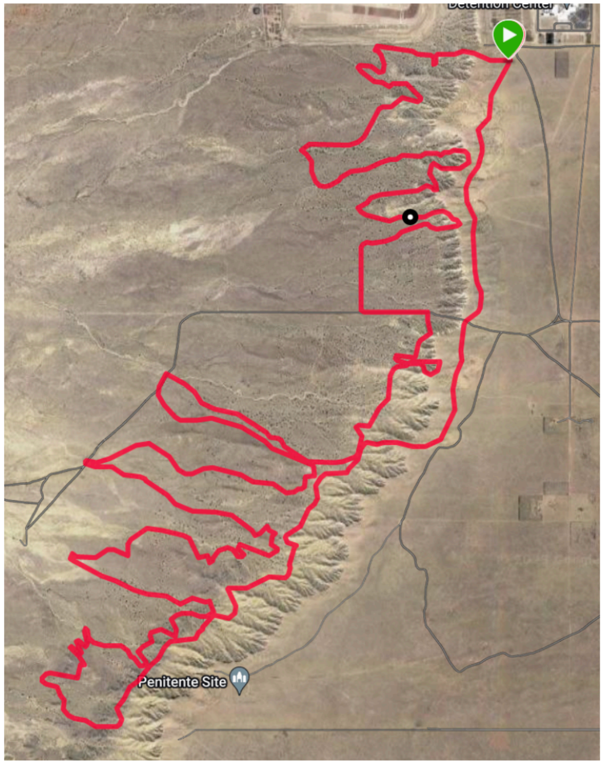
Rael San Fratello

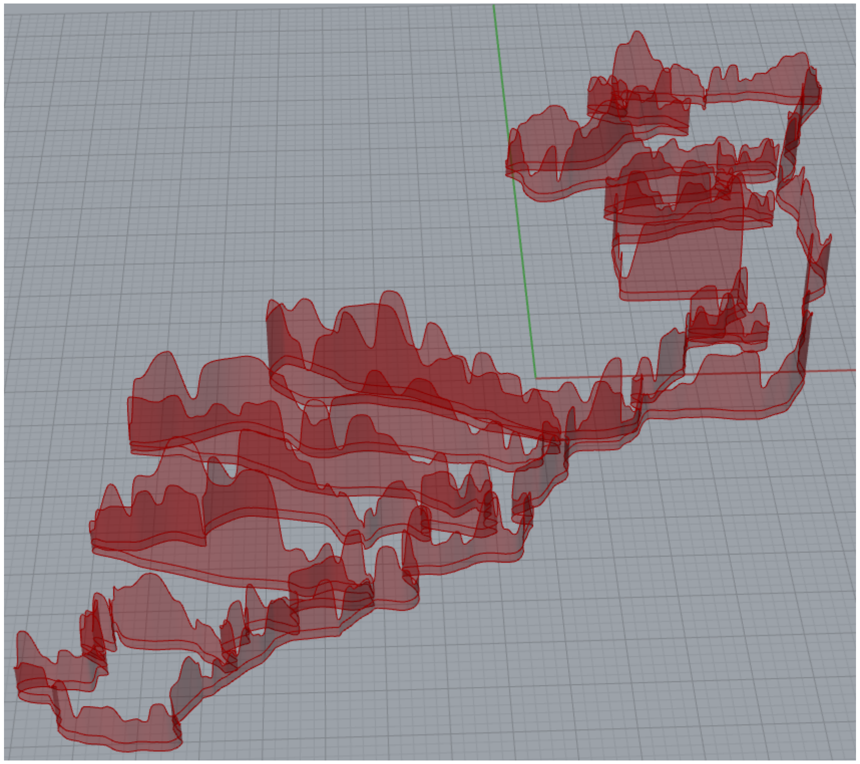
https://www.instagram.com/p/CyCTaWrP9ig/?img_index=1

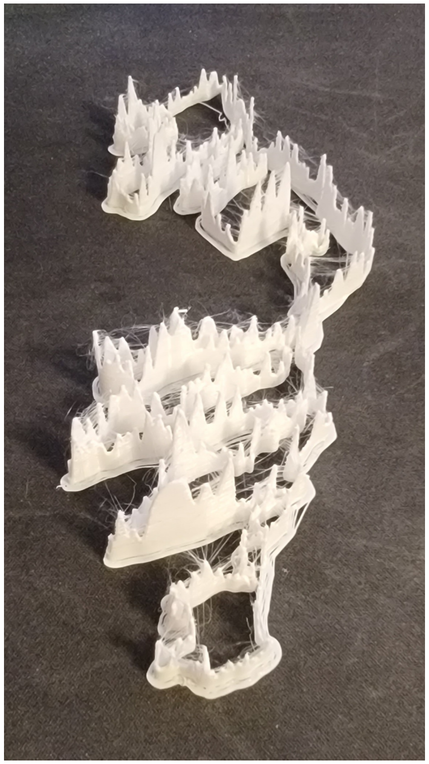
# Final Project Proposals

https://handandmachine.org/classes/computational_fabrication/2024/10/03/final-project-proposal-9/

# Dirt Bike Rides Data Physicalization



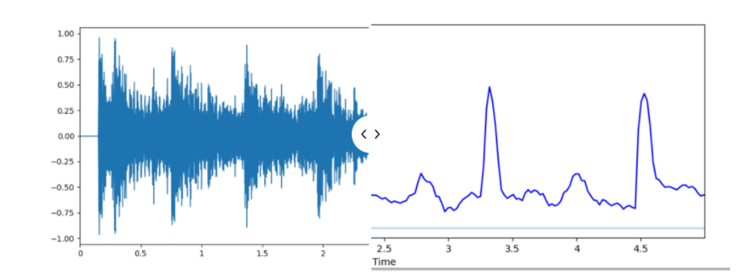




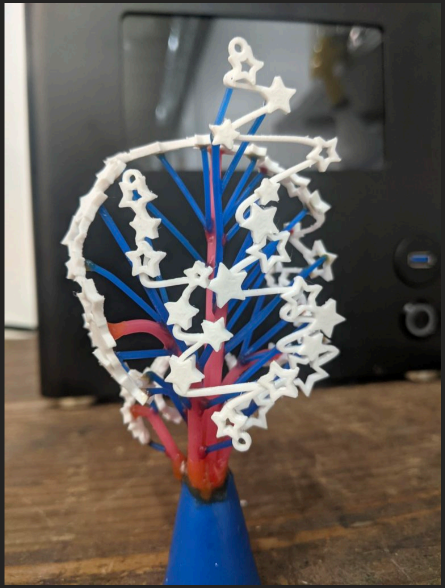
https://handandmachine.org/classes/computational_fabrication/2023/12/07/andy-thornhill-final-project-presentation/

Andy Thornhill

# Music + Signal Processing + Jewelry



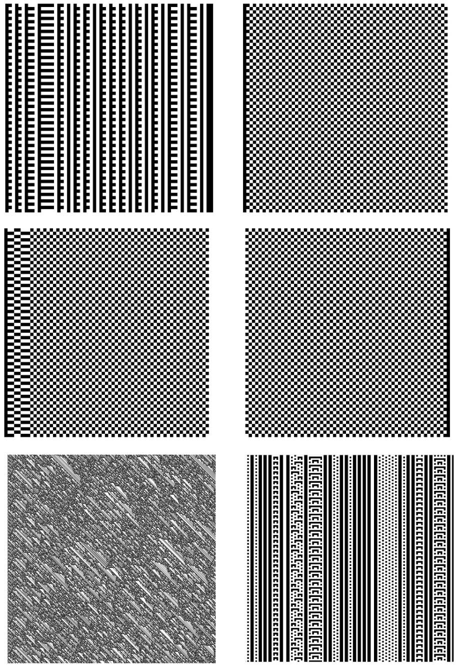
Left: Signal without Feature Extraction, Right: Feature Extraction using Spectral Centroids

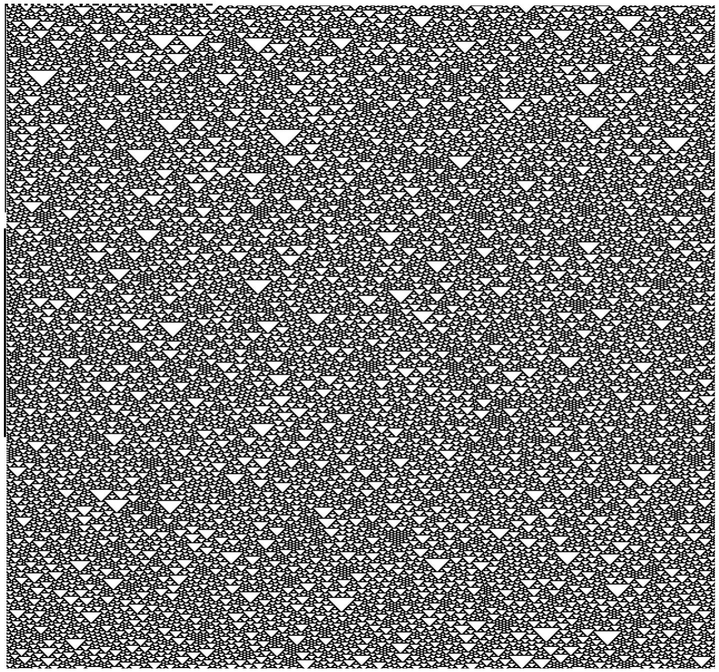


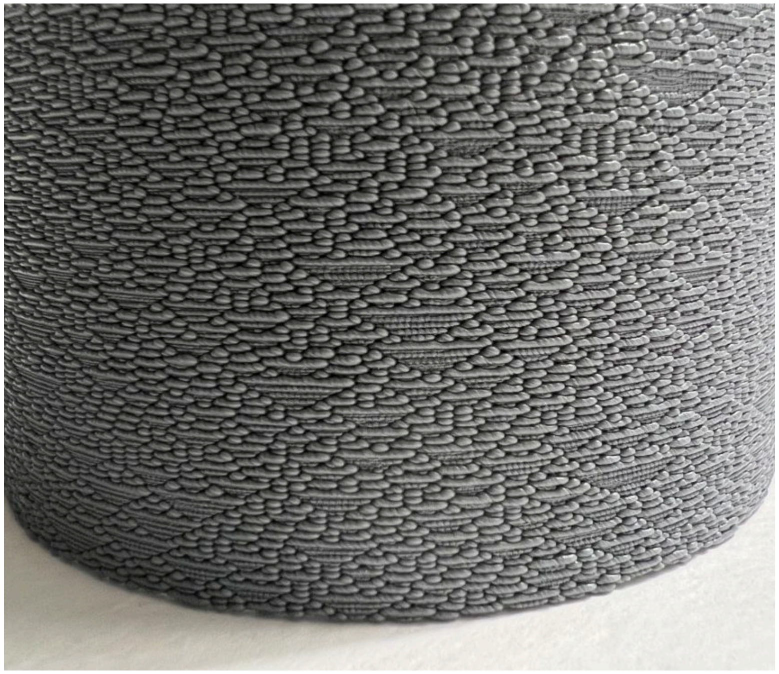
Michelle Louie

Michelle Louie

# 1D Cellular Automata & Genetic Algorithms



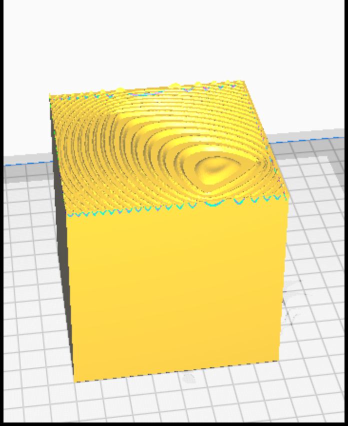


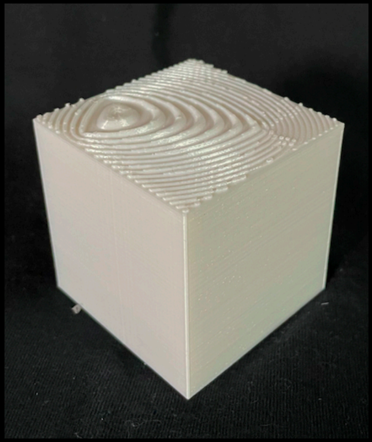


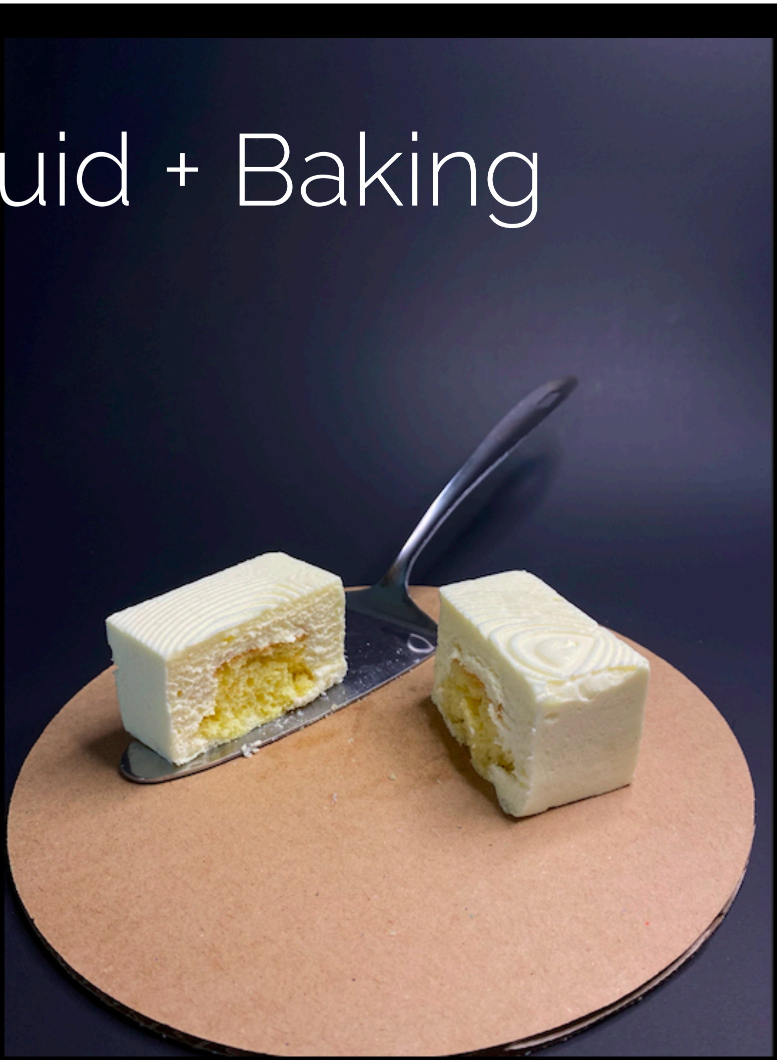
https://handandmachine.org/classes/computational_fabrication/2022/05/10/final-project-abraham-dominguez-hernandez/

Abraham Fernandez

# Modeling Ripples on Fluid + Baking

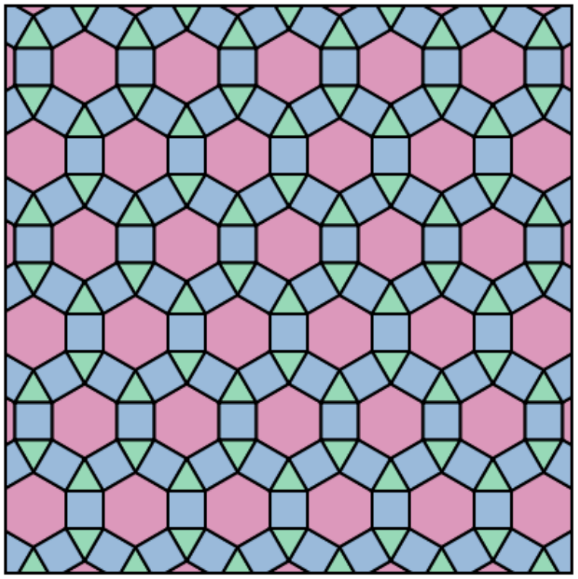






Amber Sustaita and Reuben Fresquez

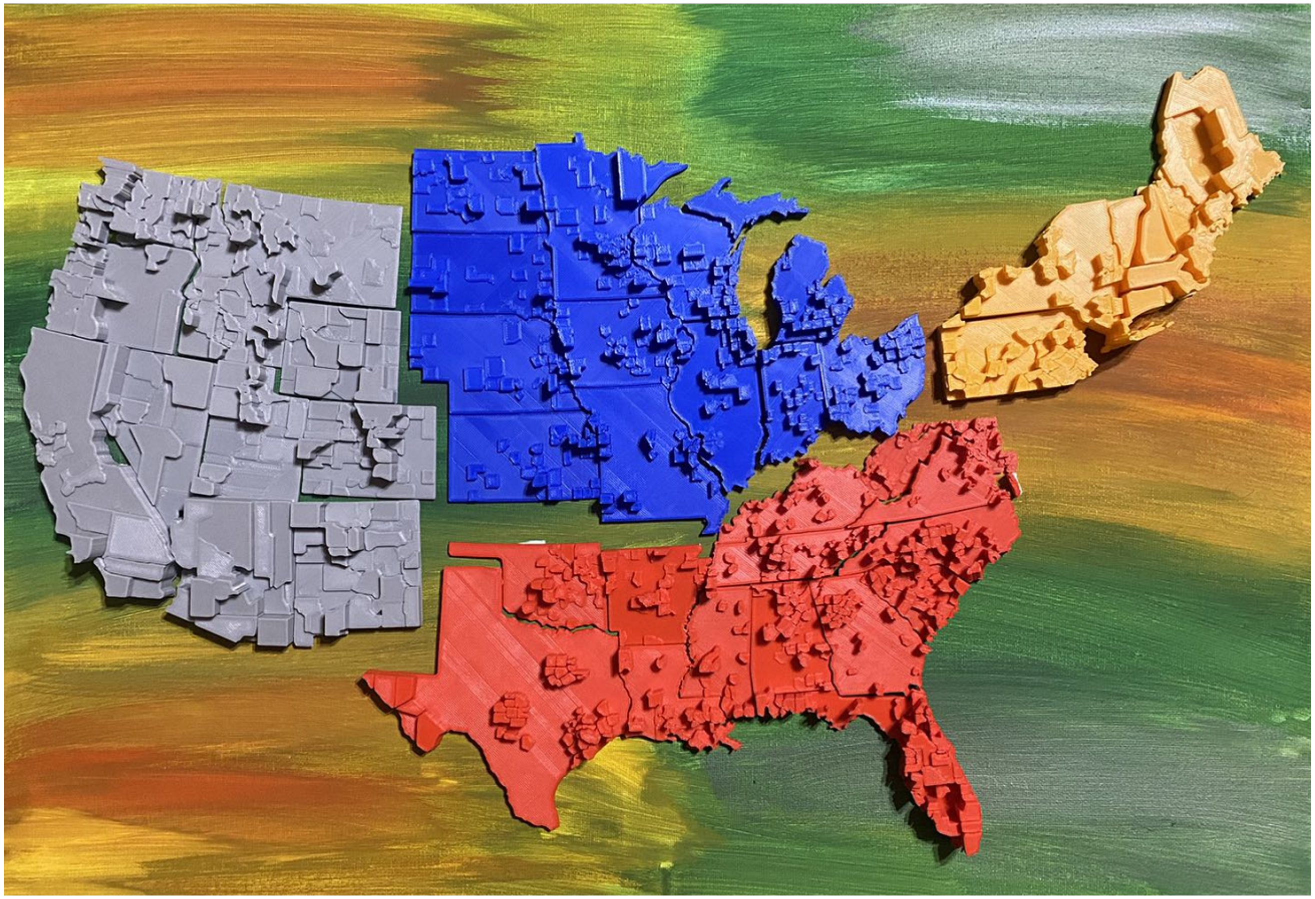
# Mathematical Tiling + Baking



https://handandmachine.org/classes/computational_fabrication/2022/05/10/jaminis-final-project/

Jamini Sahu

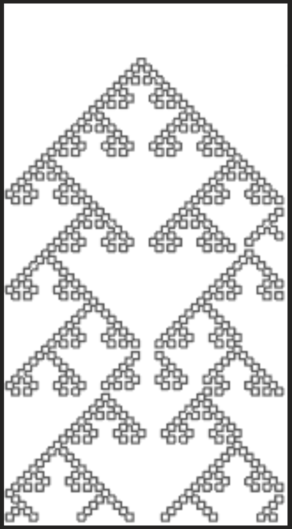
# USA Population Data by County, 3D Puzzle Map

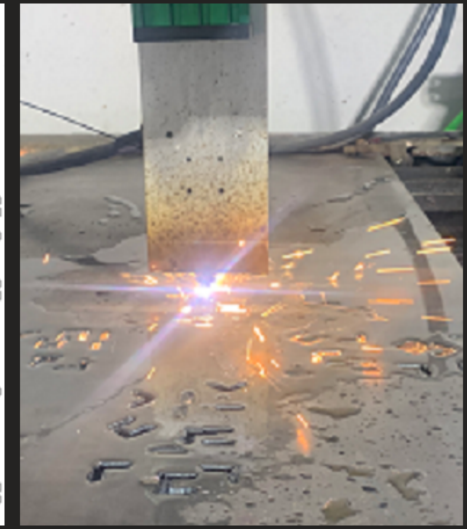


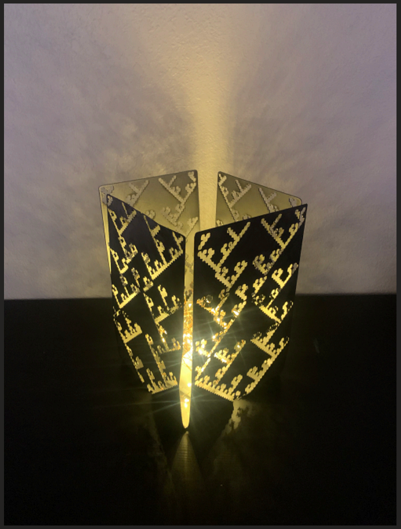
https://handandmachine.org/classes/computational_fabrication/2023/12/07/lin-and-jingbos-final-projects/

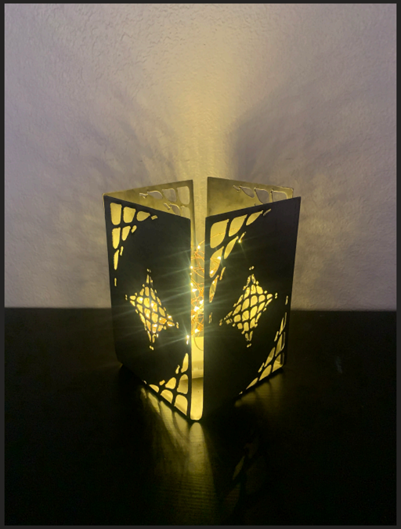
Jinbo Liang and Lin Liu

# Metal Lamps via Plasma Cutting



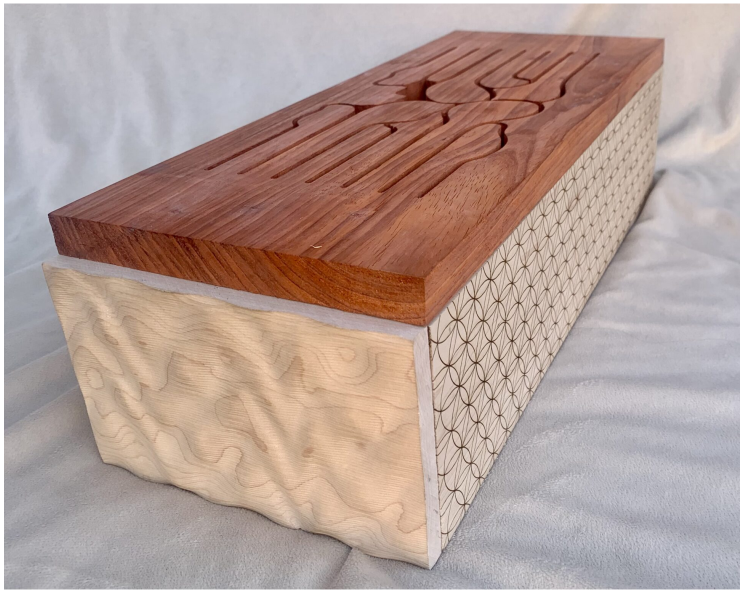


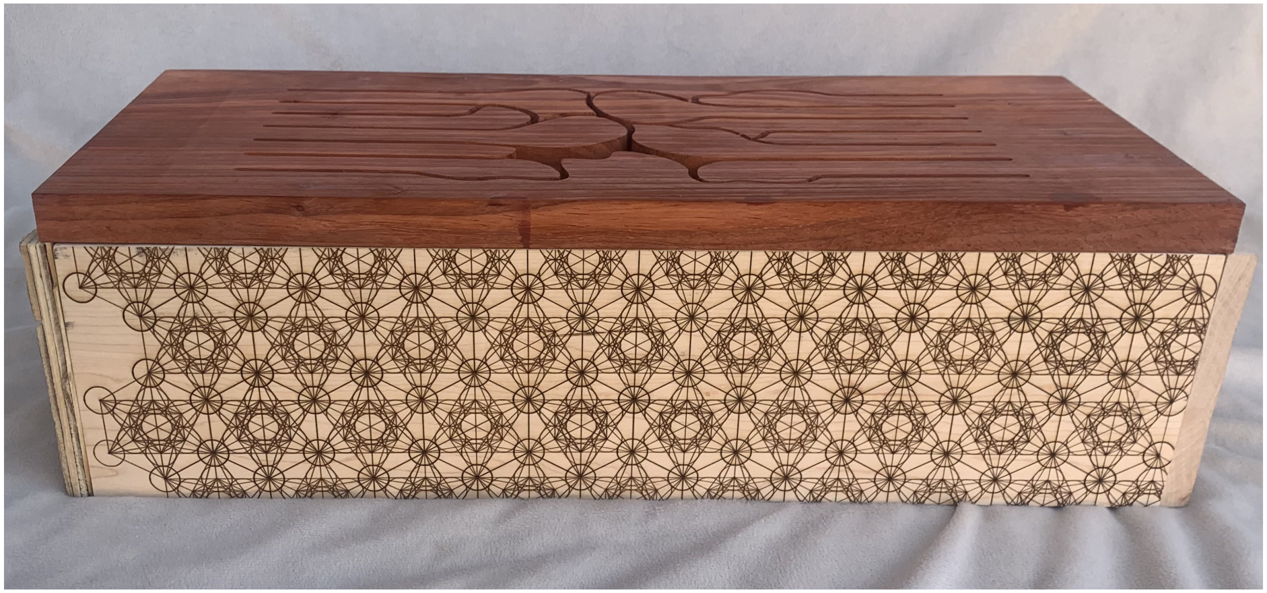




Alyshia Bustos

# Wood CNC: Wooden Tongue Drum





https://handandmachine.org/classes/computational_fabrication/2023/12/07/final-project-9/

Lauren Urenda

# Some Possibilities

- Computationally focused exploration
- Fabrication with other machines from my lab, your home, or across campus:
  Laser cutters, CNC machines, SLA 3D printer, dual nozzle 3D printer, ceramic 3D printer, embroidery machine, knitting machine
- Use a fabrication service:
  3D print in metal, nylon, transparent plastic, etc.
  https://jlc3dp.com/
  Design and fabricate custom woven,  knit, or printed fabric:
  https://www.wovns.com/
  https://www.knitwise.com/collections/knitwise-design-guide
  https://www.spoonflower.com/
- Experiment with different materials and/or processes

# Large Assignment 4: G-Code

questions?

# Today: Slicers

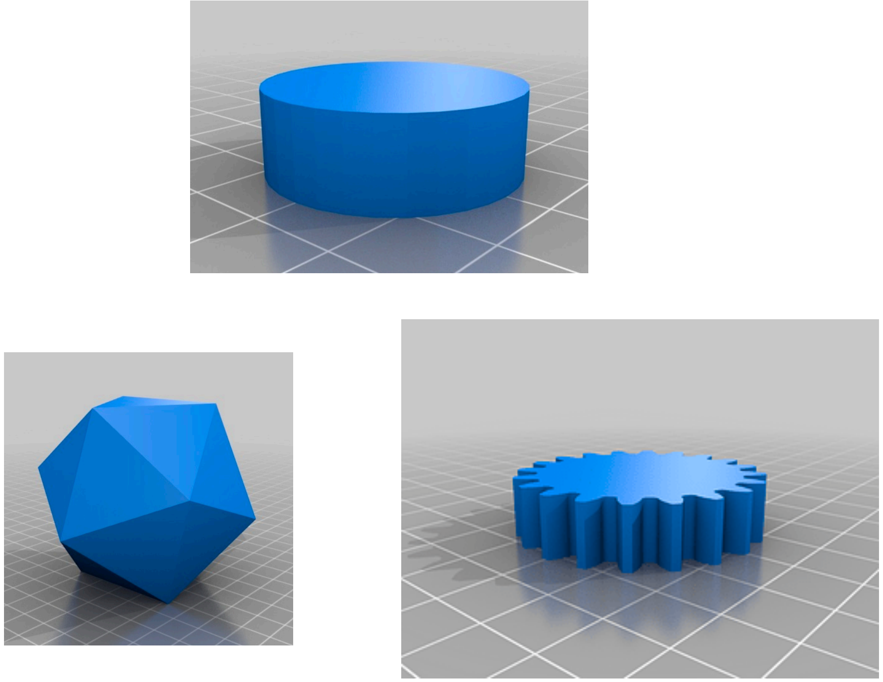
# Slicer

- Takes an arbitrary geometry/shape as input
- Generates a toolpath (.gcode file) that will 3D print the shape
- Steps:
  - Slice shape into horizontal layers
  - For each layer, generate a toolpath
  - Toolpath for a layer may include walls, infill, and support

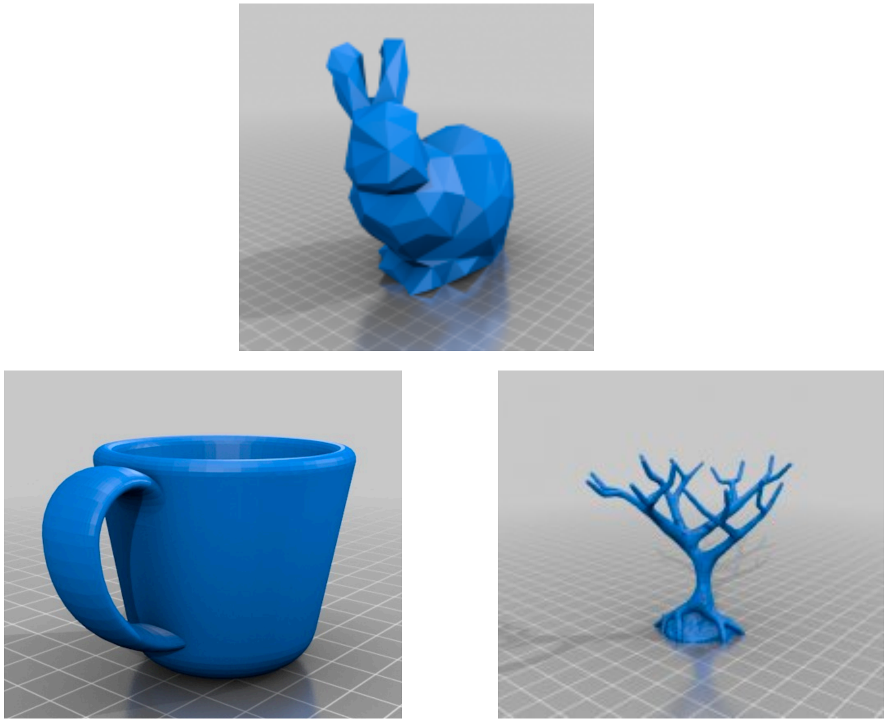
# What We’ll Build: Simplest Slicer

- Generates a toolpath (.gcode file) that will traverse the outside wall of simple solids.
- Limitations on input shapes
  - Simple topology (no holes)
  - Simple geometry: each slice of shape must be a single surface
- Steps:
  - Slice shape into horizontal layers
  - For each layer, generate a toolpath that follows the outside curve of the shape

# What We’ll Build: Simplest Slicer



can slice

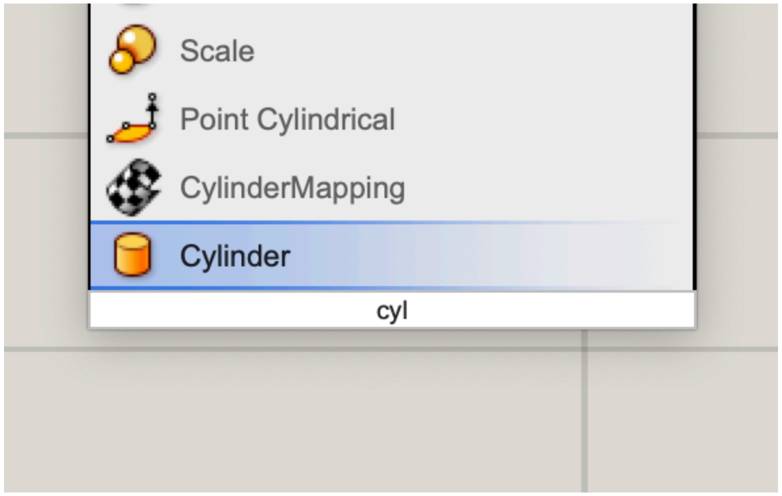


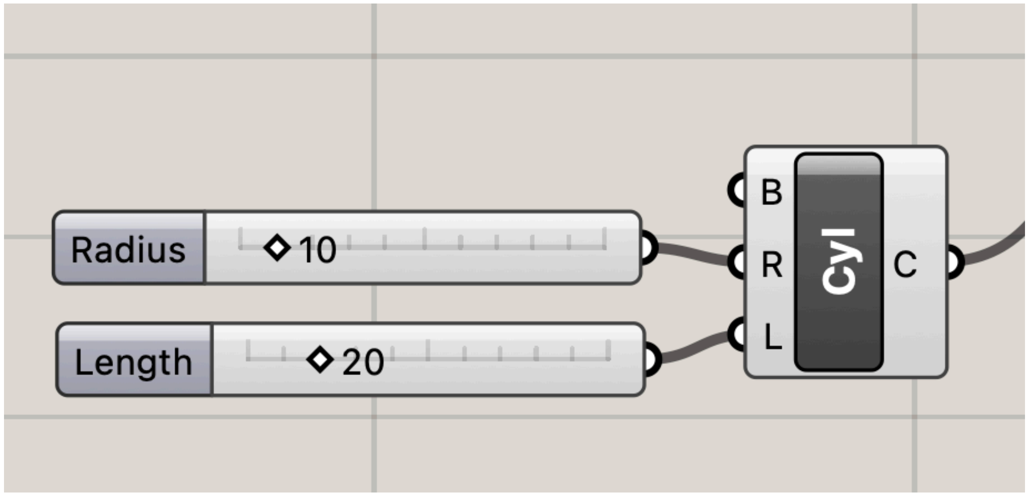
can't slice

questions?

# Open Rhino and Grasshopper

# Create a Cylinder

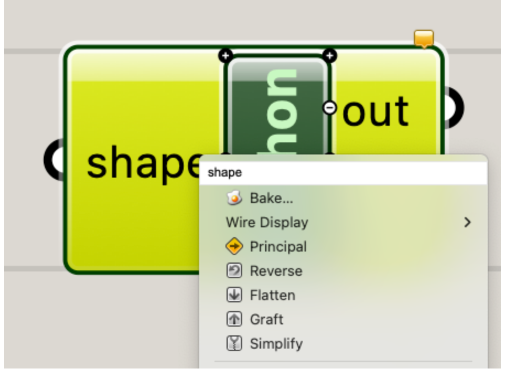




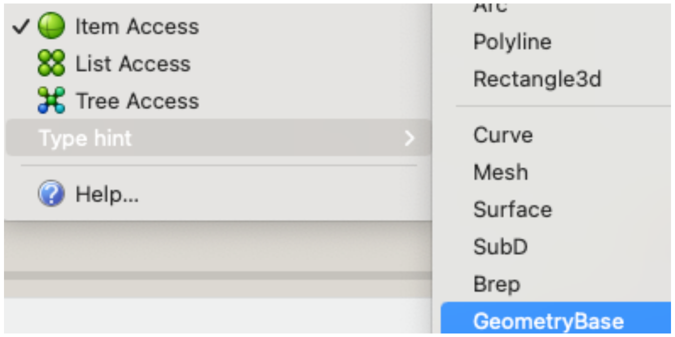
# Code Overview

1. Get the height of the shape using BoundingBox.
2. Slice shape using AddSrfCountorCurves. This function outputs a list of edge curves.
3. Break each edge curve into a list of points using DivideCurve.
4. Follow this list of points with a turtle using set_position_point.

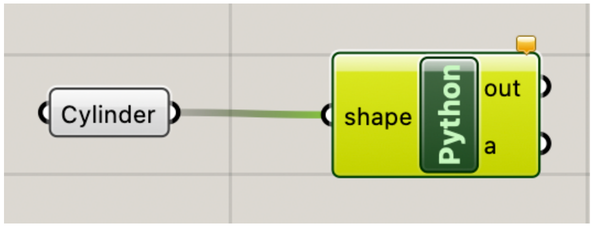
# Implementation



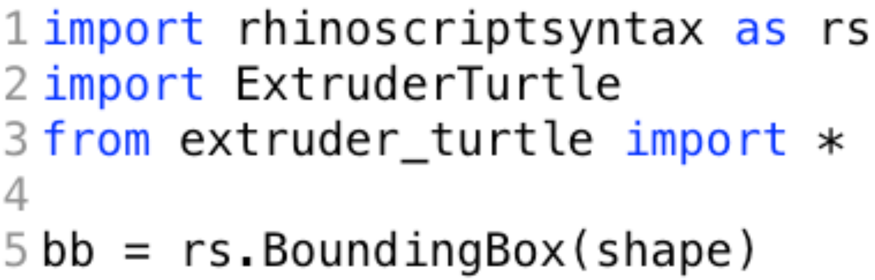
Python block with one input, name it shape



Type hint —> GeometryBase

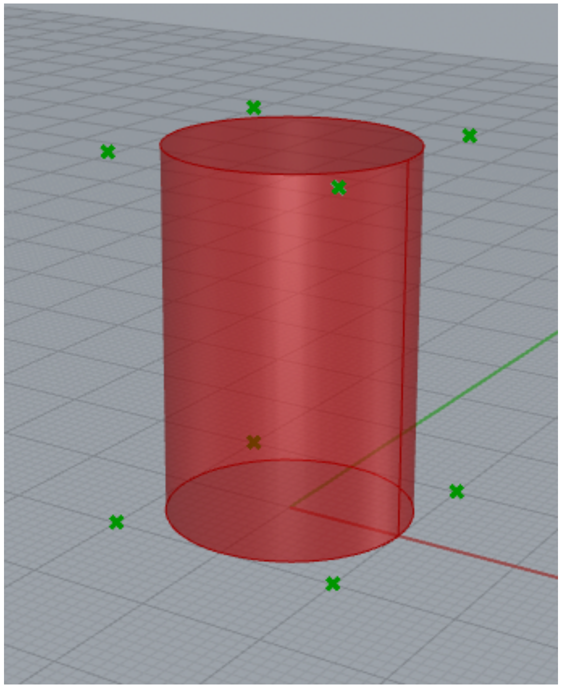


# BoundingBox(shape)

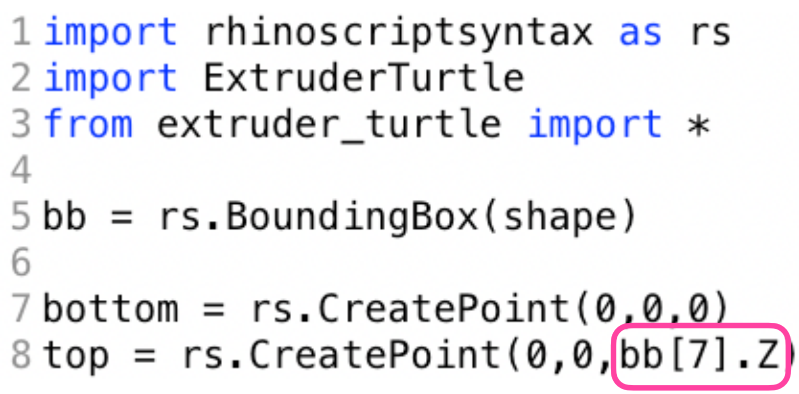


```python
import rhinoscriptsyntax as rs
import ExtruderTurtle
from extruder_turtle import *

bb = rs.BoundingBox(shape)
```

returns a list of 8 points that define a bounding box

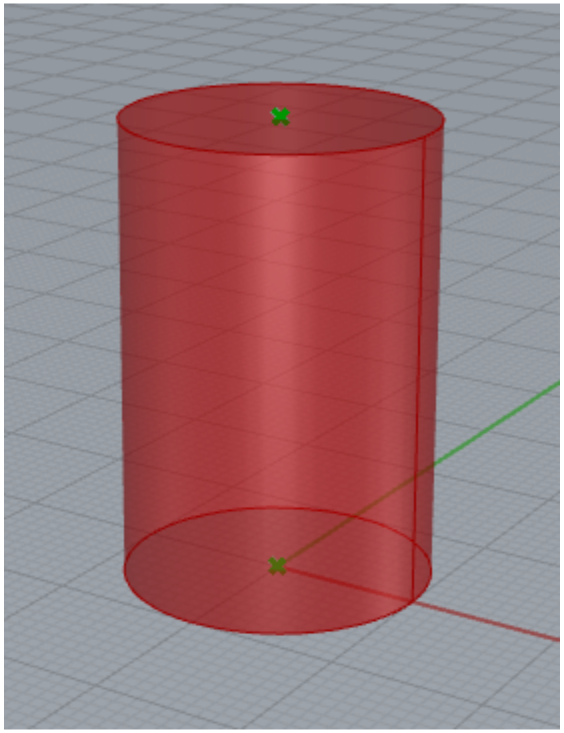


# Get top and bottom points of shape

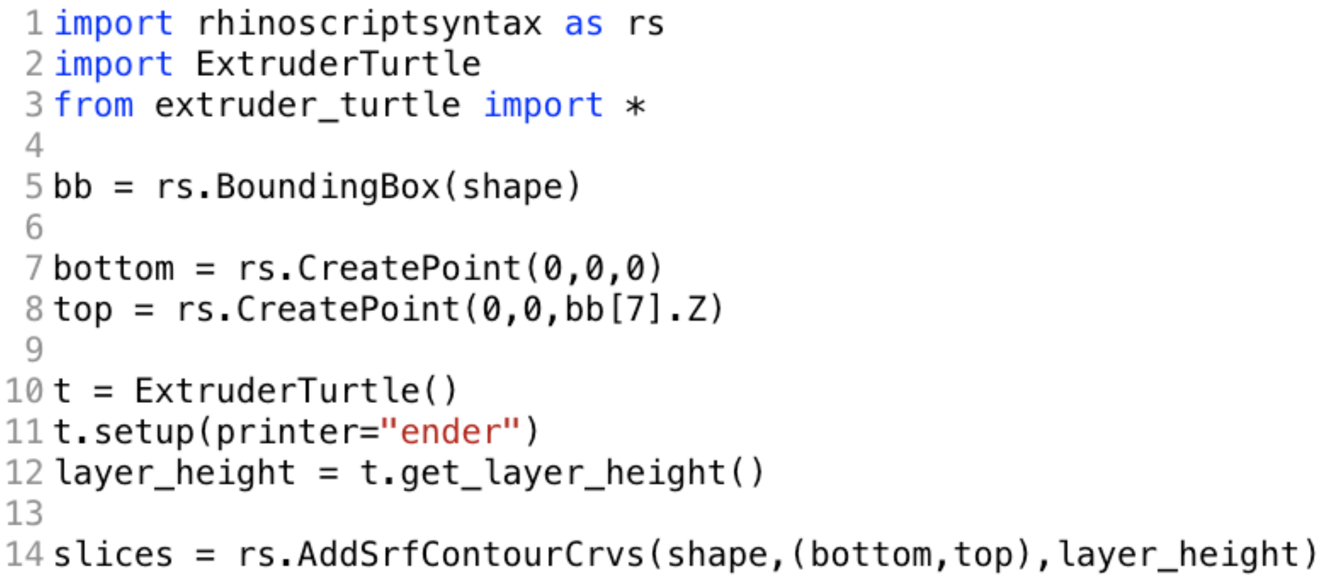


```
import rhinoscriptsyntax as rs
import ExtruderTurtle
from extruder_turtle import *

bb = rs.BoundingBox(shape)

bottom = rs.CreatePoint(0,0,0)
top = rs.CreatePoint(0,0,bb[7].Z)
```

use Bounding Box to find Z coordinate of top point

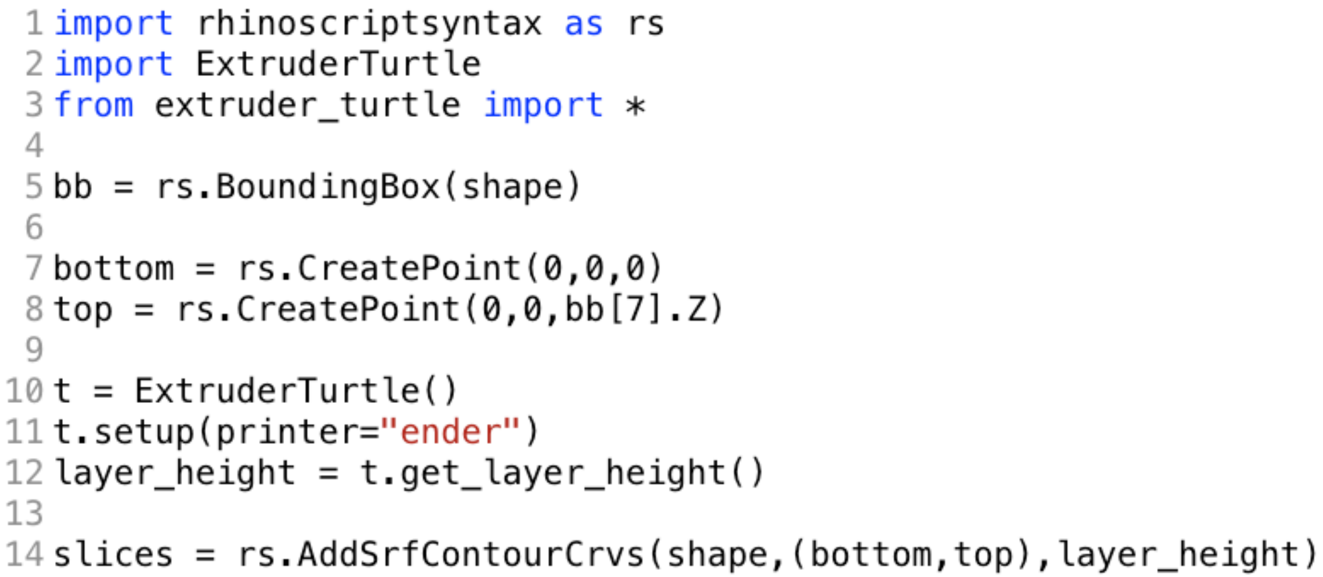


# Set up Turtle

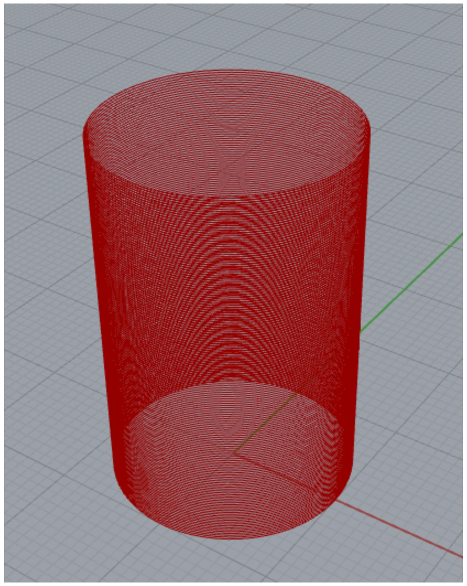


```
import rhinoscriptsyntax as rs
import ExtruderTurtle
from extruder_turtle import *

bb = rs.BoundingBox(shape)

bottom = rs.CreatePoint(0,0,0)
top = rs.CreatePoint(0,0,bb[7].Z)

t = ExtruderTurtle()
t.setup(printer="ender")
layer_height = t.get_layer_height()

slices = rs.AddSrfContourCrvs(shape,(bottom,top),layer_height)
```

# Slice shape!



```python
import rhinoscriptsyntax as rs
import ExtruderTurtle
from extruder_turtle import *

bb = rs.BoundingBox(shape)

bottom = rs.CreatePoint(0,0,0)
top = rs.CreatePoint(0,0,bb[7].Z)

t = ExtruderTurtle()
t.setup(printer="ender")
layer_height = t.get_layer_height()

slices = rs.AddSrfContourCrvs(shape,(bottom,top),layer_height)
```

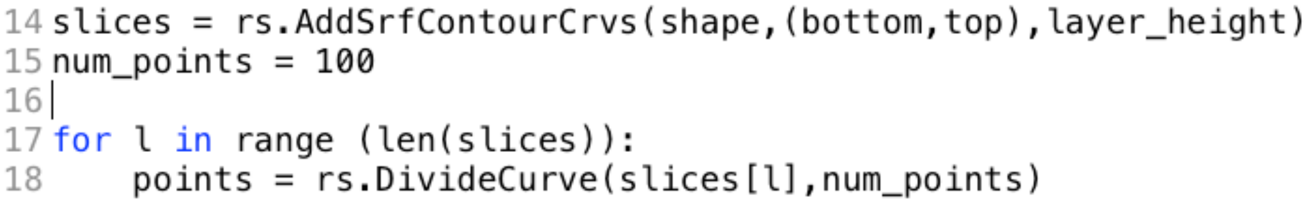


AddSrfCountourCrvs outputs a list of curves from bottom to top at intervals of layer_height

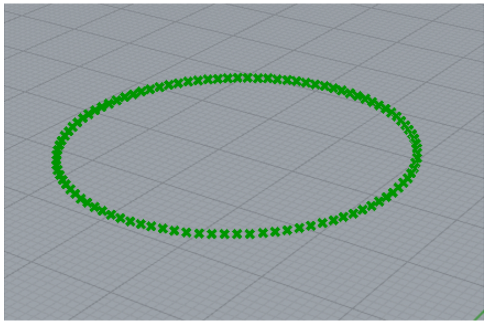
questions?

# Now we’ll create a Turtle path

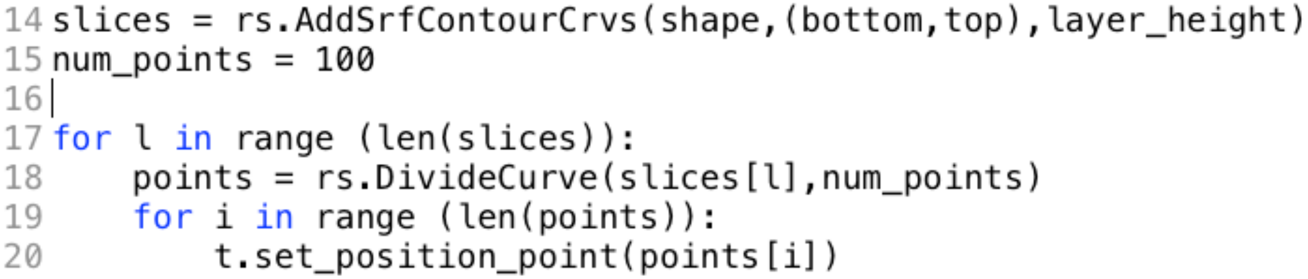
# DivideCurve: break each curve into a list of points



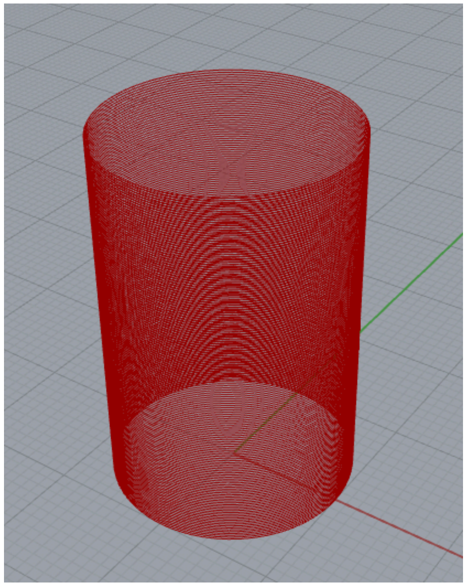
```
slices = rs.AddSrfContourCrvs(shape,(bottom,top),layer_height)
num_points = 100

for l in range (len(slices)):
    points = rs.DivideCurve(slices[l],num_points)
```



# Follow points with turtle

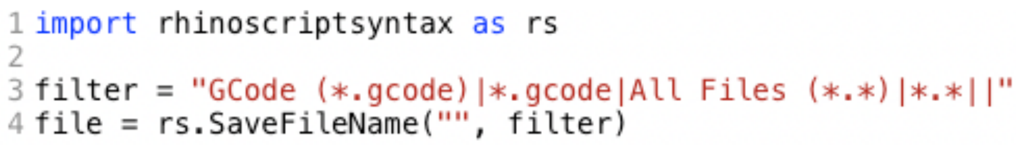


```
slices = rs.AddSrfContourCrvs(shape,(bottom,top),layer_height)
num_points = 100

for l in range (len(slices)):
    points = rs.DivideCurve(slices[l],num_points)
    for i in range (len(points)):
        t.set_position_point(points[i])
```

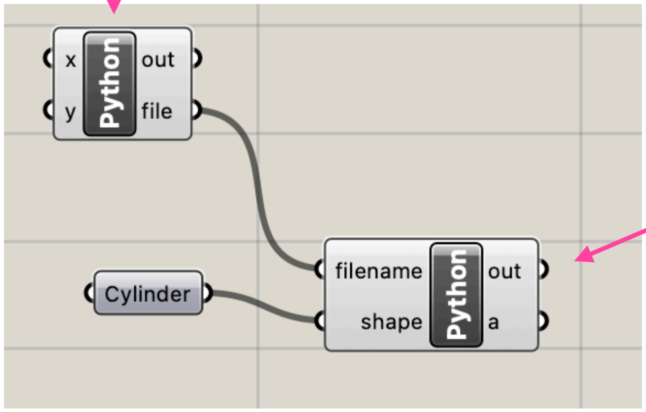


questions?

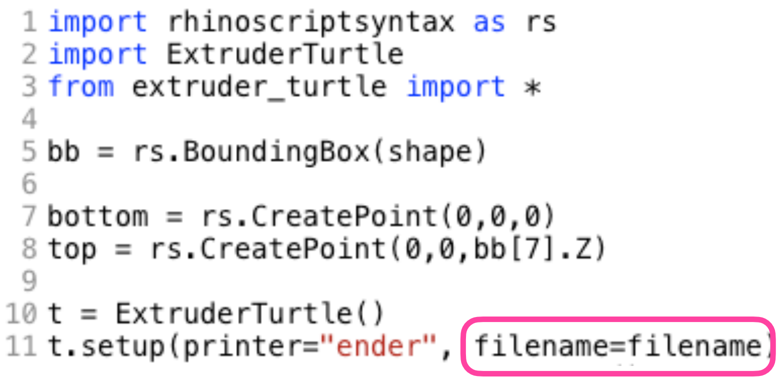
# Add file generation to code



```python
import rhinoscriptsyntax as rs

filter = "GCode (*.gcode)|*.gcode|All Files (*.*)|*.*||"
file = rs.SaveFileName("", filter)
```

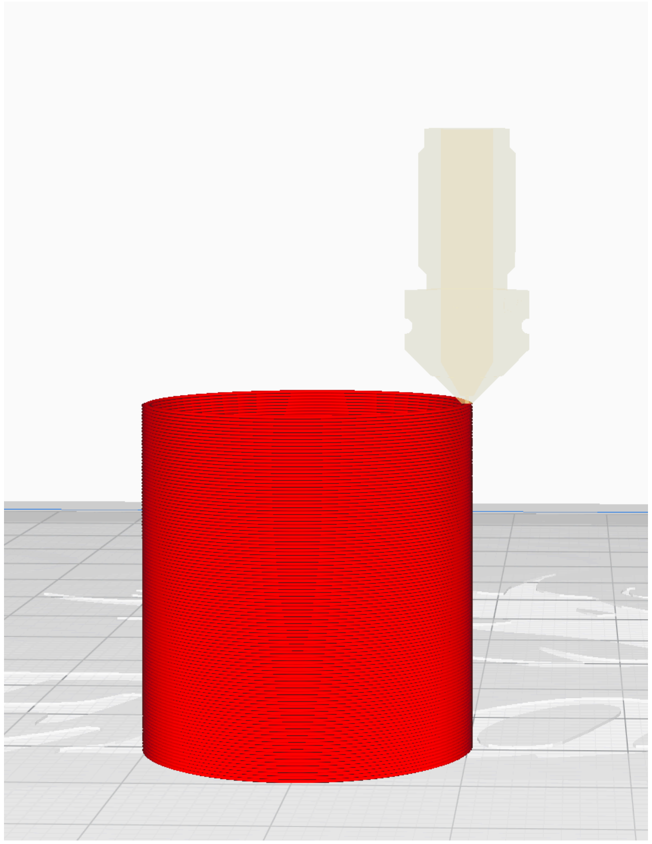


Type Hint for filename should be str



```python
import rhinoscriptsyntax as rs
import ExtruderTurtle
from extruder_turtle import *

bb = rs.BoundingBox(shape)

bottom = rs.CreatePoint(0,0,0)
top = rs.CreatePoint(0,0,bb[7].Z)

t = ExtruderTurtle()
t.setup(printer="ender", filename=filename)
```

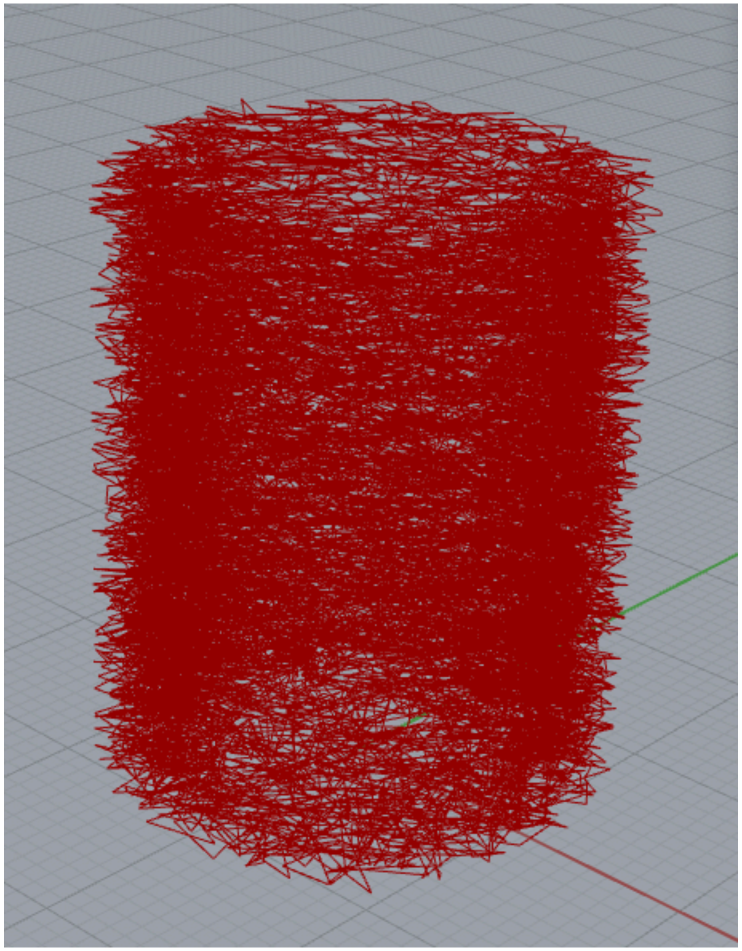
# Preview in Cura



Now, for the fun part!

How can we make this slicer interesting?

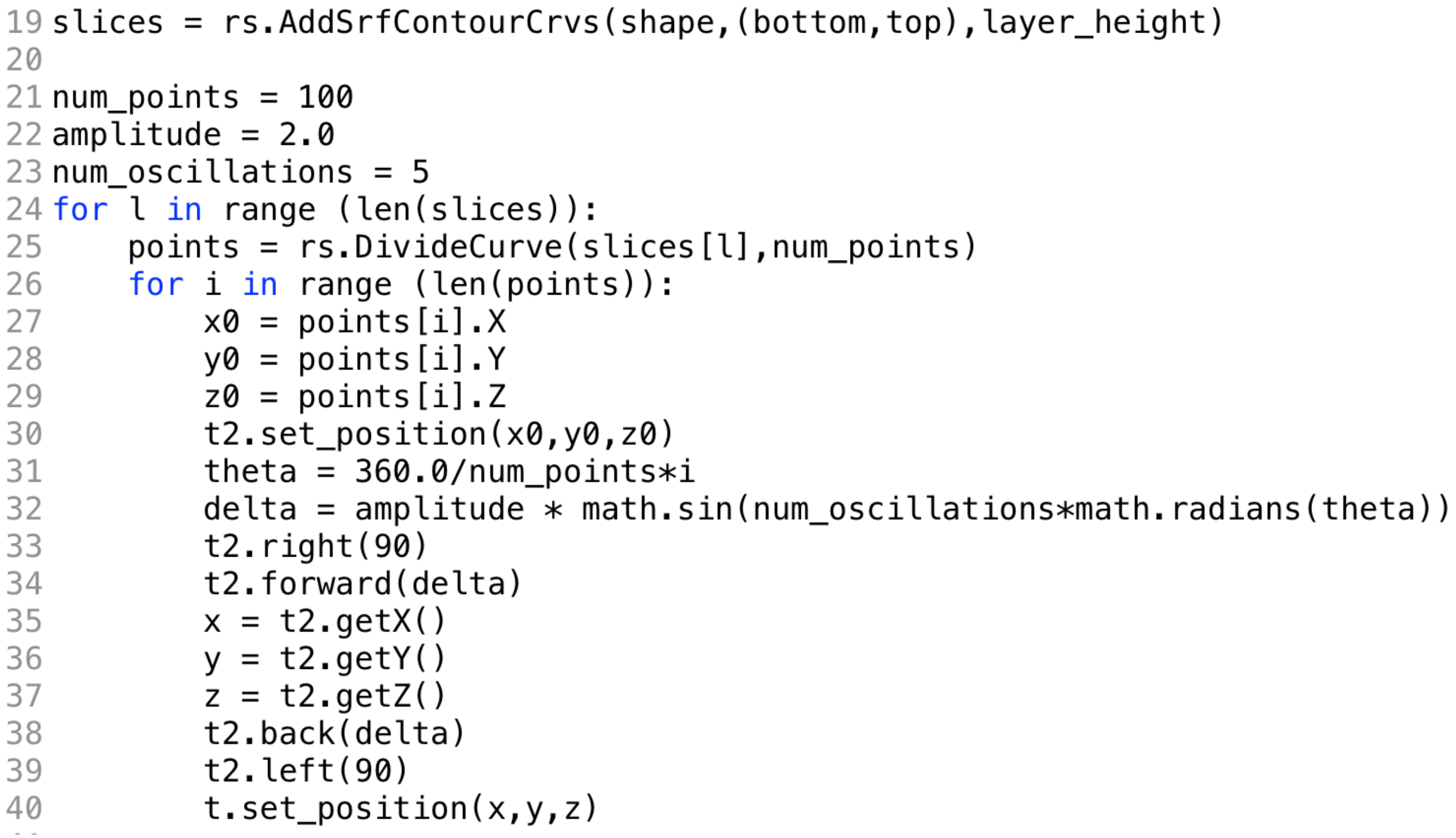
# A little randomness



# The magic of multiple Turtles

- Use one turtle to generate interesting points that are based on the slice curve for each layer. This turtle might generate a bunch of extraneous lines that you don't want to include in your print
- Use a second (primary) turtle to follow only the points that you want to include in your toolpath.

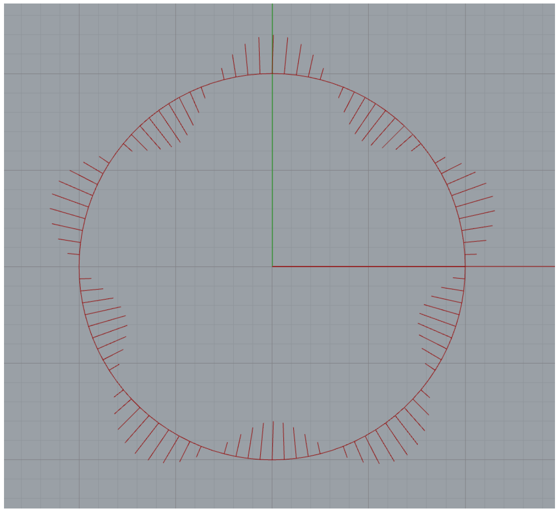
# Two turtle example code



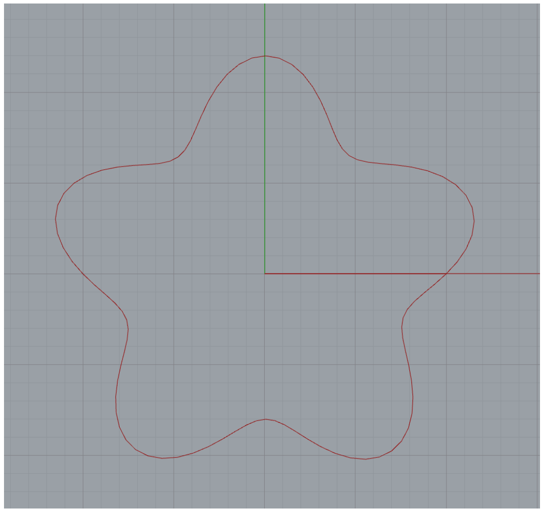
```
slices = rs.AddSrfContourCrvs(shape,(bottom,top),layer_height)

num_points = 100
amplitude = 2.0
num_oscillations = 5
for l in range (len(slices)):
    points = rs.DivideCurve(slices[l],num_points)
    for i in range (len(points)):
        x0 = points[i].X
        y0 = points[i].Y
        z0 = points[i].Z
        t2.set_position(x0,y0,z0)
        theta = 360.0/num_points*i
        delta = amplitude * math.sin(num_oscillations*math.radians(theta))
        t2.right(90)
        t2.forward(delta)
        x = t2.getX()
        y = t2.getY()
        z = t2.getZ()
        t2.back(delta)
        t2.left(90)
        t.set_position(x,y,z)
```

# Output

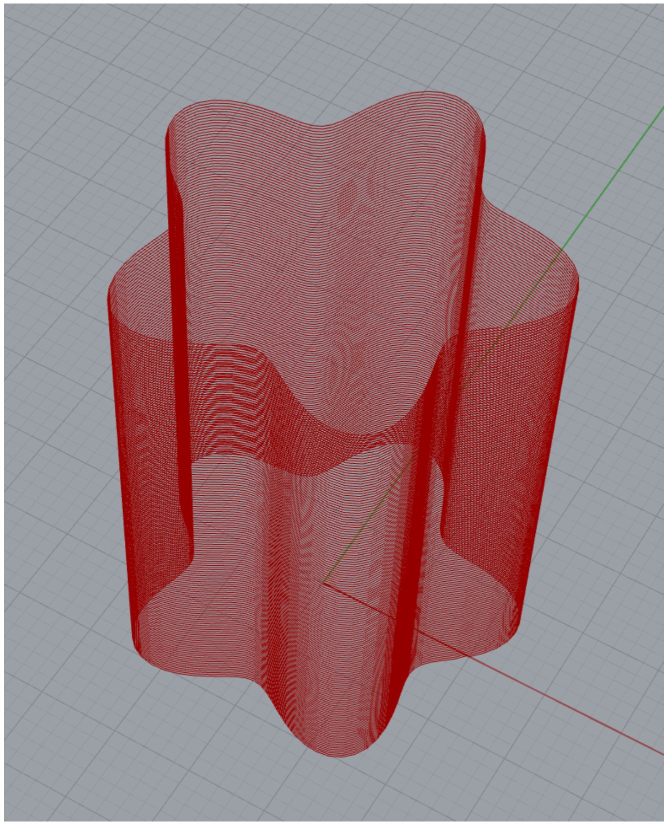
top view



t2 path



t path



t path

questions?

# Thank you!

CS 491 and 591
Professor: Leah Buechley
https://handandmachine.cs.unm.edu/classes/Computational_Fabrication_Spring2021/