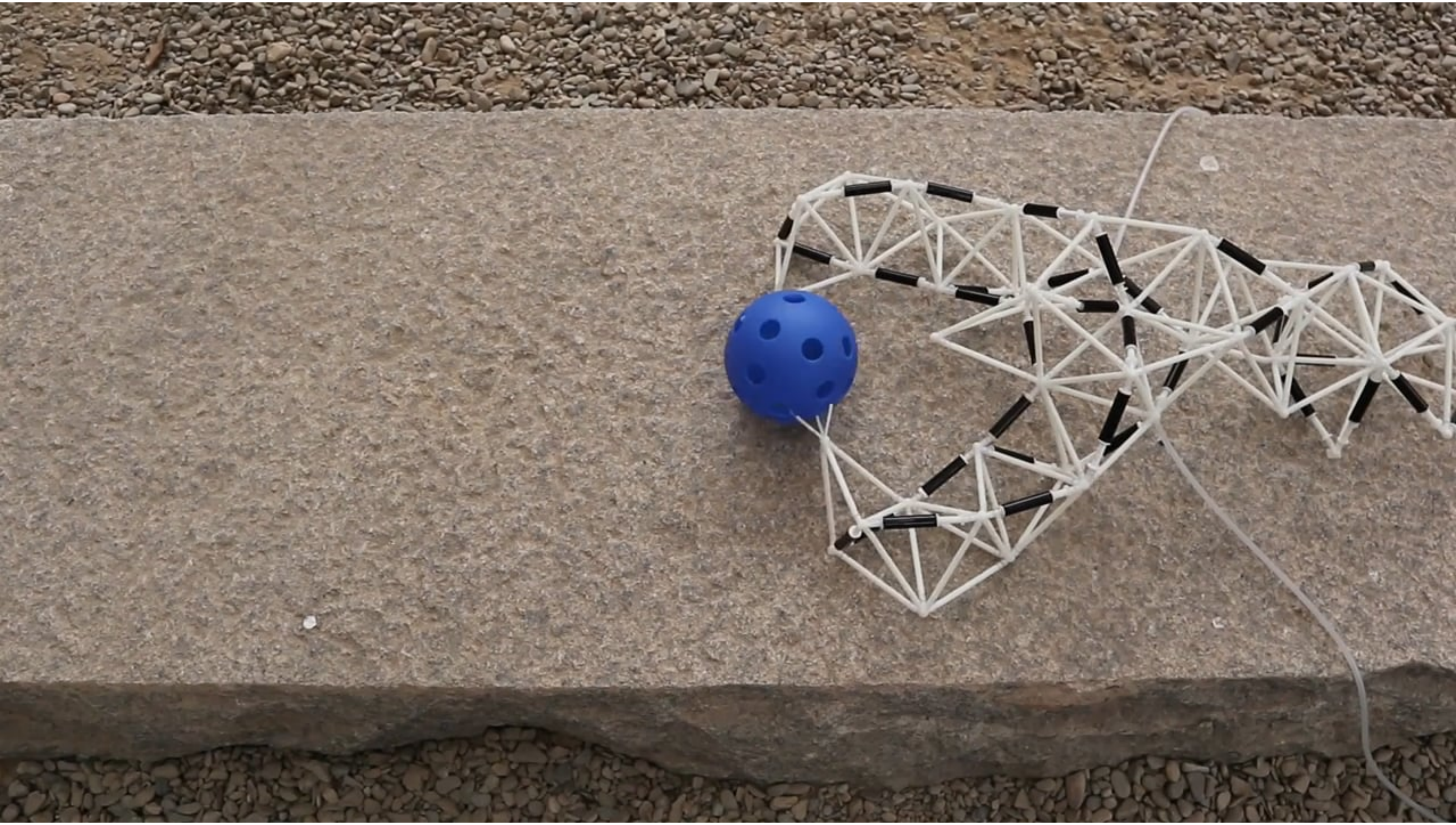
Computational Fabrication

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

CS Researcher (Designer): Lining Yao

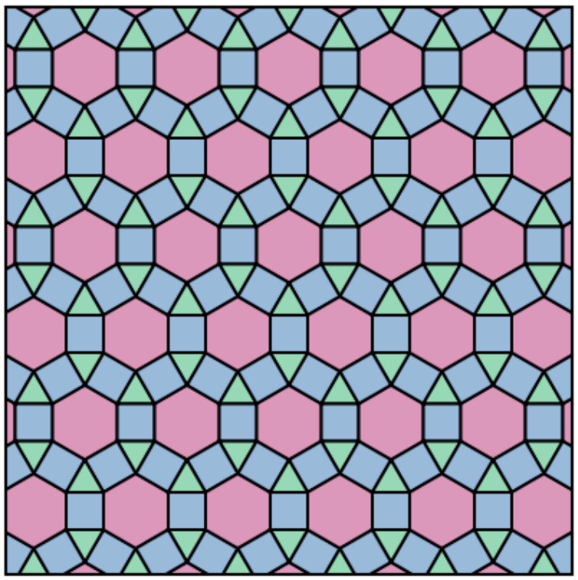
https://www.morphingmatter.cs.cmu.edu/



Final Project Proposals

https://handandmachine.org/classes/computational_fabrication/2023/10/10/final-project-proposal-4/

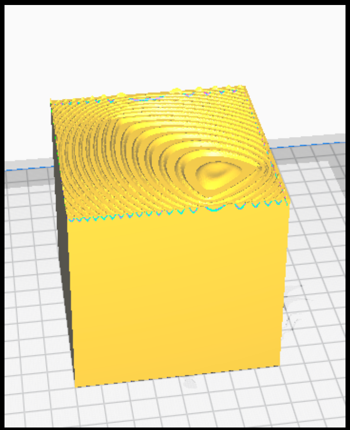
Mathematical Tiling + Baking

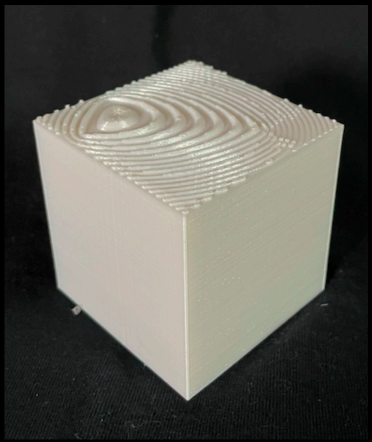
  

https://handandmachine.org/classes/computational_fabrication/2022/05/10/jaminis-final-project/

Jamini Sahu

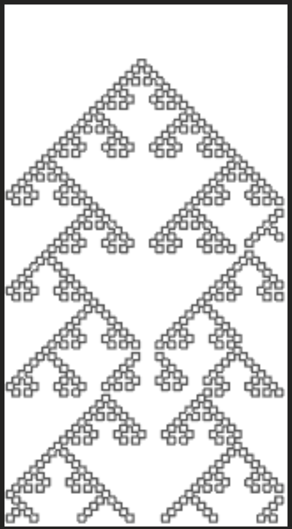
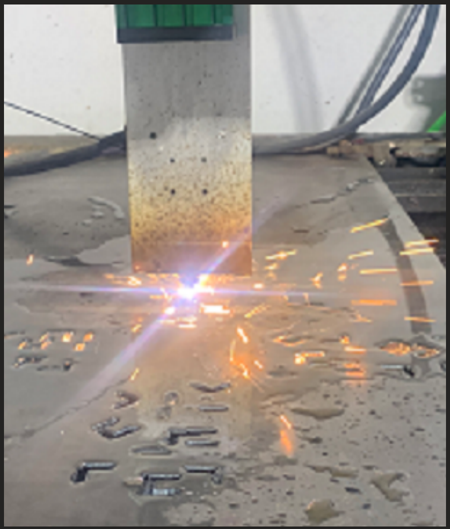
Modeling Ripples on Fluid + Baking

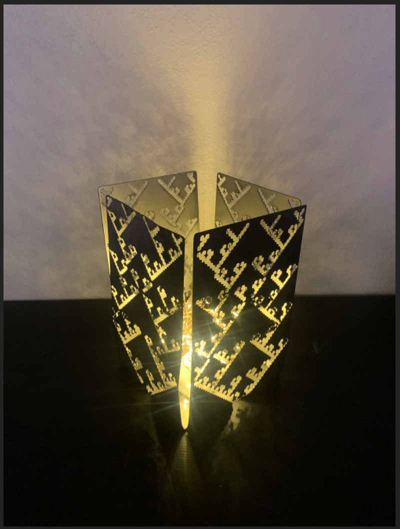




Amber Sustaita and Reuben Fresquez

Metal Lamps via Plasma Cutter

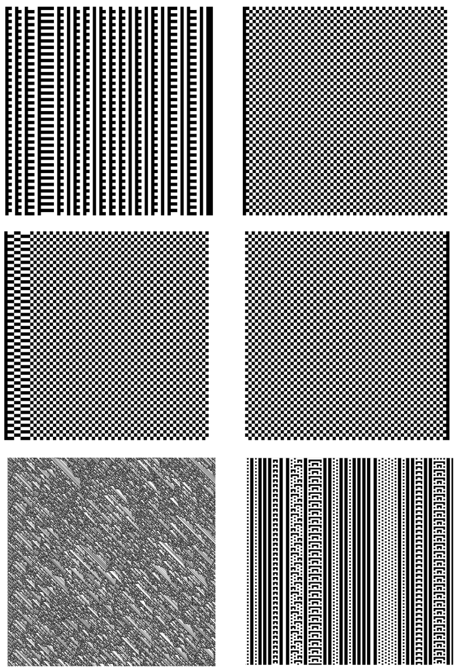


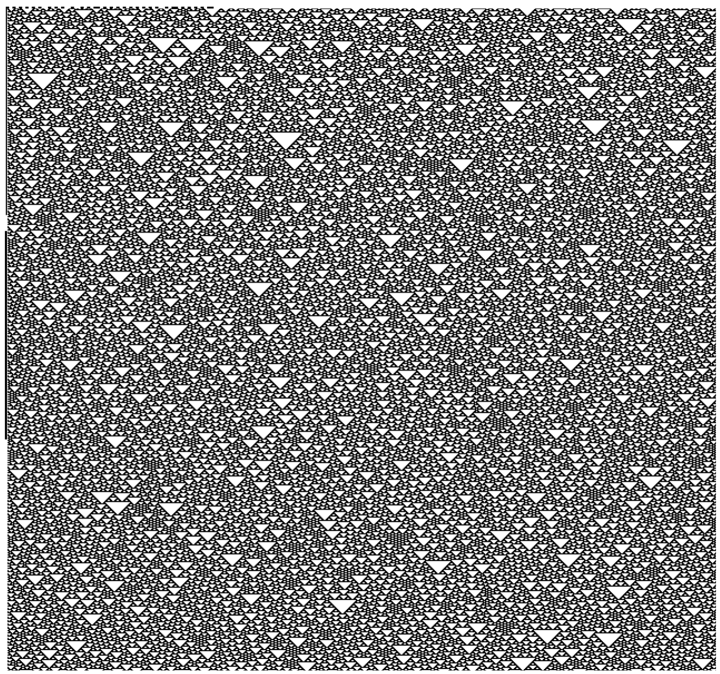

Alyshia Bustos

Chess Game

Kai Vallon & Chris Shelton

1D Cellular Automata & Genetic Algorithms

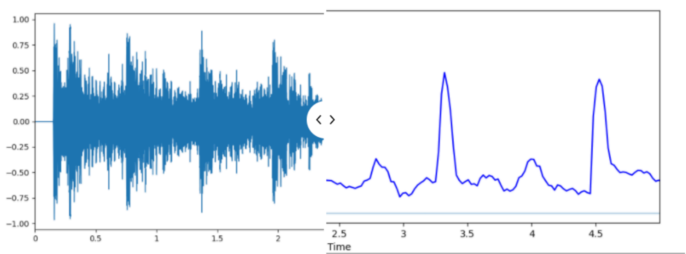




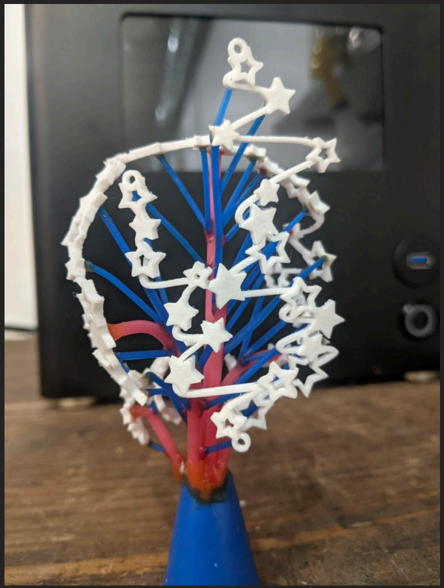
https://handandmachine.org/classes/computational_fabrication/2022/05/10/final-project-abraham-dominguez-hernandez/

Abraham Fernandez

Music + Signal Processing + Jewelry



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



Michelle Louie

Michelle Louie

questions?

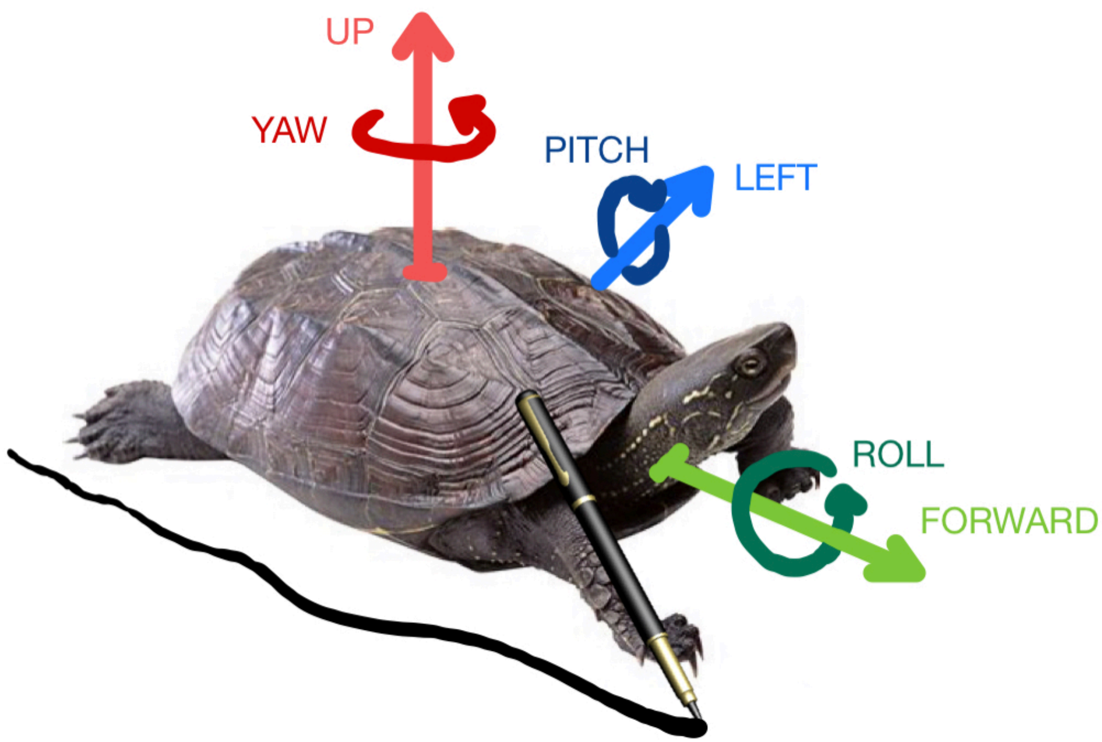
Large Assignment 4: G-Code

Due on Halloween (Tuesday October 31)
Create three objects by generating gcode
https://handandmachine.org/classes/computational_fabrication/2023/10/19/gcode/

questions?

Today: GCODE
Extruder Turtle Library

Extruder Turtle Library



https://handandmachine.org/projects/extruder_turtle_rhino/

Turtle generates a 3D printed path as it moves by generating g-code

Functionality: Movement

- **t.forward**(distance) moves the turtle forward by a given distance, extruding along the way if the pen is down.
- **t.left**(theta) turns the turtle left by a given angle. This is an alias for t.yaw(theta).
- **t.right**(theta) turns the turtle right by a given angle. This is an alias for t.yaw(-theta).
- **t.pitch_up**(theta) tilts the turtle "upwards" in the direction its eyes would point. Alias for t.pitch(theta).
- **t.pitch_down**(theta) tilts the turtle "downwards". Alias for t.pitch(-theta).
- **t.roll_left**(theta) rolls the turtle towards its left side. Alias for t.roll(-theta).
- **t.roll_right**(theta) rolls the turtle towards its right side. Alias for t.roll(theta).
- **t.lift**(height) lifts the turtle up by a given height. Usually used to move from one layer of a print to the next.

- **t.penup**() lifts the pen up. No extrusion will occur until it is put down again.
- **t.pendown**() puts the pen down. Extrusion will occur at a constant rate with each movement unless the pen is lifted up.

https://handandmachine.org/projects/extruder_turtle_rhino/

Functionality: Setup

- The constructor **t = ExtruderTurtle()** takes no arguments and creates a new turtle

- **t.set_extrude_rate**(extrude_rate) sets the density of extrusion, or the rate at which filament is extruded, measured in millimeters of filament per millimeters of movement.
- **t.set_speed**(speed) sets the "feedrate" or speed of the extruder.

- **t.setup()** writes the sequence of initialization commands to the g-code file (which moves the nozzle to its starting position, heats the bed and extruder, and so on). Optional arguments allow you to customize the setup process:
  x=0 is the starting x-value
  y=0 is the starting y-value
  feedrate=100 is the starting feedrate/speed
  hotend_temp=215 is the default hotend temperature
  bed_temp=60 is the default bed temperature
- **t.finish**() carries out the finalization sequence (movesthe extruder upwards, cools the bed and extruder, etc).

Functionality: GH/Rhino

- **t.draw_turtle**() generates a triangular surface that shows you the position and orientation of the Turtle in 3D space

- **t.get_lines()** generates a list of lines that allow you to visualize the path of the turtle in Rhino

GCode file structure

Header (supplied by library):
- home extruder
- heat up bed and nozzle
- extrude lines along edge

Main code (generated by turtle movement):
- move extruder to start point
- build shape with G1 commands
- E commands determine amount of filament extruded

Footer (supplied by library):
- return home
- turn off heaters and fans

questions?

Experiment with the 3D Turtle

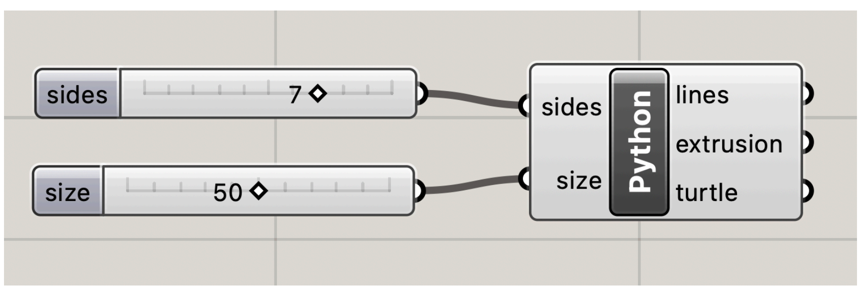
Drag out a Python scripting block and add the import statements below.

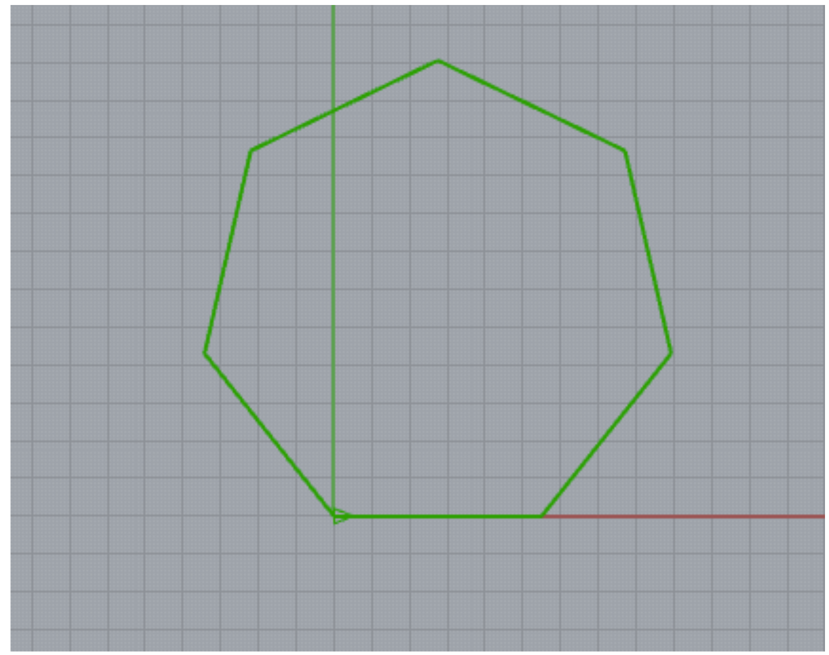
```
import extruder_turtle
from extruder_turtle import *
```

Check to make sure this compiles. If it doesn't something went wrong with installation. Go through the steps again.

Draw Polygons

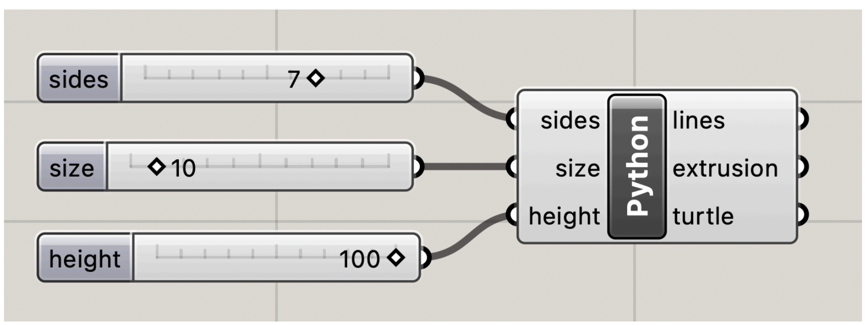
```
import rhinoscriptsyntax as rs
import extruder_turtle
from extruder_turtle import *

t = ExtruderTurtle()

for i in range (0,sides):
    t.forward(size)
    t.left(360/sides)

lines = t.get_lines()
turtle = t.draw_turtle()
```

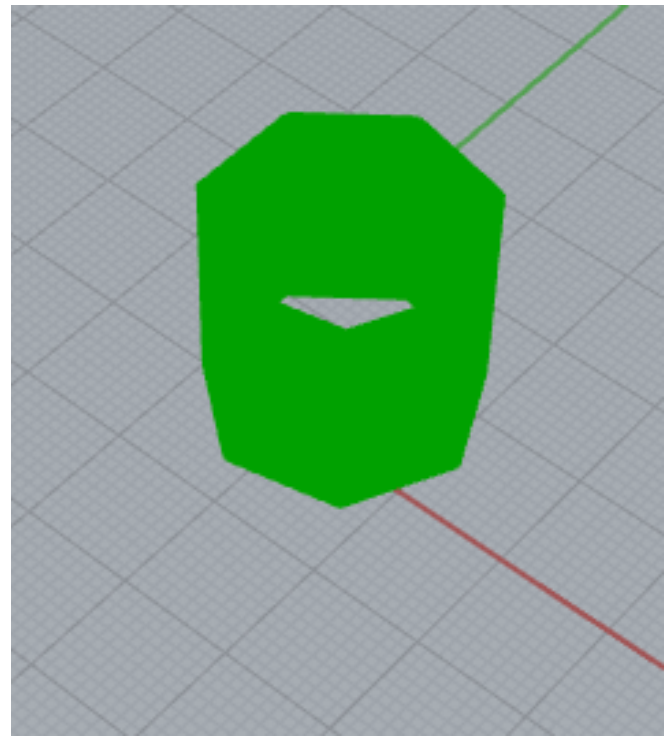




Draw Prisms

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

t = ExtruderTurtle()

for h in range (0,height):
    for i in range (0,sides):
        t.forward(size)
        t.left(360/sides)
    t.lift(.2)

lines = t.get_lines()
turtle = t.draw_turtle()
```

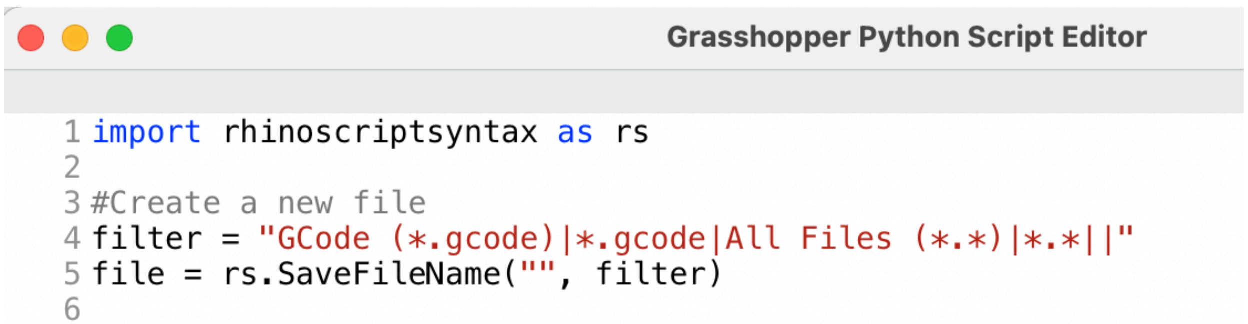




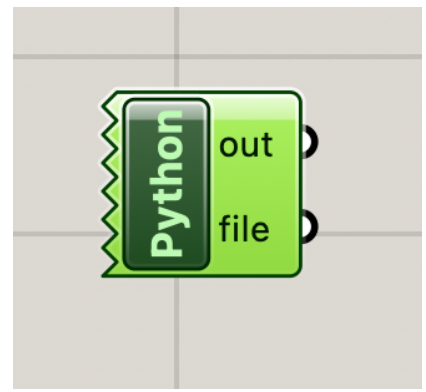
Turtle to GCode

Files in Grasshopper

Drag out a new Python scripting block and add the statements below: https://handandmachine.org/classes/computational_fabrication/2022/03/29/code-to-save-a-gcode-file/

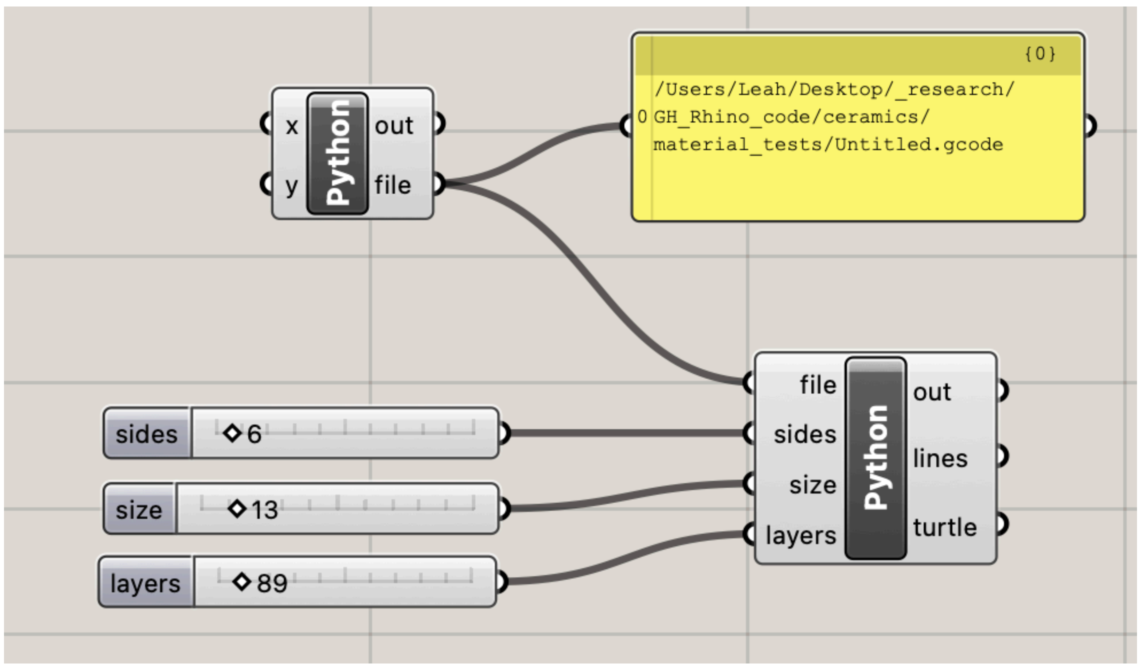


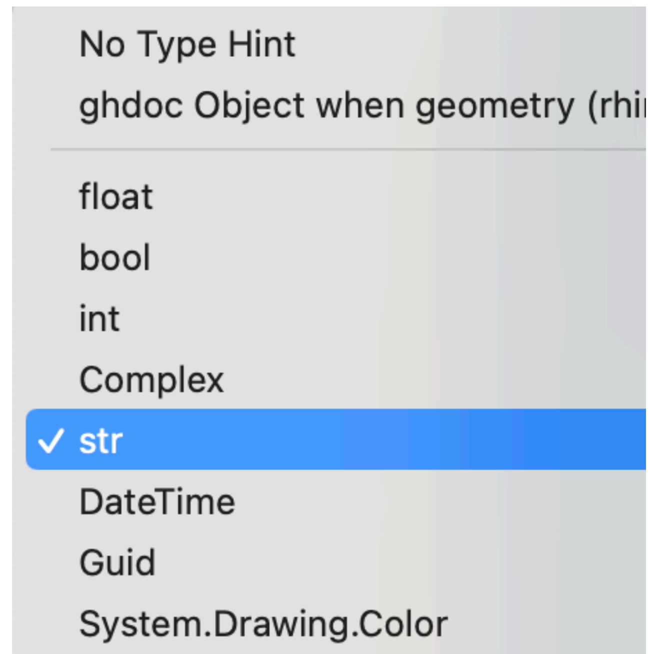

```
import rhinoscriptsyntax as rs

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



change the name of the output to “file” to match code

Choose a name for your generated file
Connect to your other Python block





choose **str** as the Type Hint for the “file” input

Turtle to GCode: Add Lines

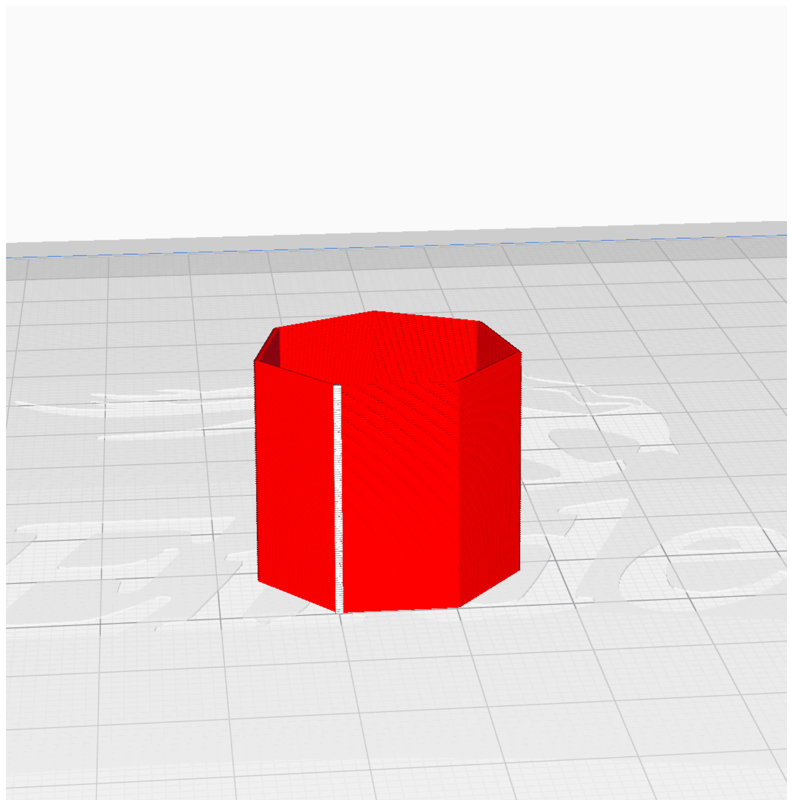
```
import rhinoscriptsyntax as rs
import extruder_turtle
from extruder_turtle import *

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

for h in range (0,height):
    for i in range (0,sides):
        t.forward(size)
        t.left(360/sides)
    t.lift(.2)

lines = t.get_lines()
turtle = t.draw_turtle()
t.finish()
```

Look at the G-Code you generated

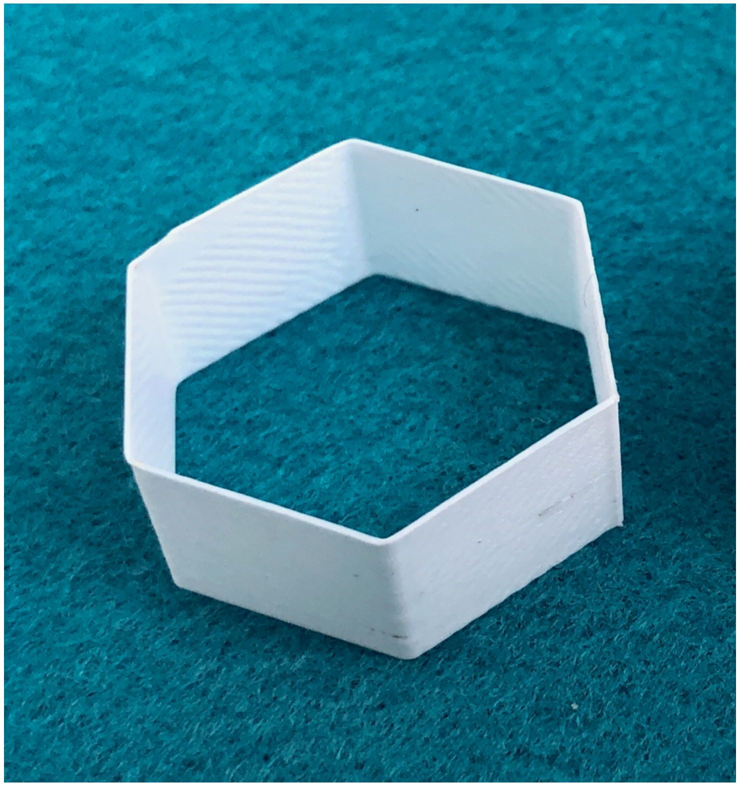
```
; ############### begin header for Ender ##############
G92 E0 ; Reset Extruder
G28 ; Home all axes
G90 ; Absolute coordinates for X,Y,Z
M190 S60 ; Set bed temperature and wait
M109 S205 ; Set extruder temperature and wait
G1 Z2.0 F3000 ; Move Z to 2mm above bed
G1 X0.1 Y20 Z0.3 F5000.0 ; Move to start position
G1 X0.1 Y200.0 Z0.3 F1500.0 E15 ; Draw the first line
G1 X0.4 Y200.0 Z0.3 F5000.0 ; Move to side a little
G1 X0.4 Y20 Z0.3 F1500.0 E30 ; Draw the second line
G92 E0 ; Reset Extruder
G1 Z2.0 F3000 ; Move Z Axis up little to prevent scratching of Heat Bed
G1 X5 Y20 Z0.3 F5000.0 ; Move over to prevent blob squish
G92 E0 ; Reset extruder position to zero
G1 F300 E-3
G1 F2000
G1 0 0 .3               ; lift nozzle above bed a little

G1 X0 Y0 Z0       ; go to the starting position
F300 E3                 ; Extrude to get ready

G1 F1000            ; set the speed/feedrate
M83                     ; Relative extrustion
G91                     ; relative coordinates for X,Y,Z axes
; ###############  end header  ##############
```

```
G1 X10.0 Y0.0 Z0.0 E0.5
G1 X6.2349 Y7.81831 Z0.0 E0.5
G1 X-2.22521 Y9.74928 Z0.0 E0.5
G1 X-9.00969 Y4.33884 Z0.0 E0.5
G1 X-9.00969 Y-4.33884 Z0.0 E0.5
G1 X-2.22521 Y-9.74928 Z0.0 E0.5
G1 X6.2349 Y-7.81831 Z0.0 E0.5
G1 Z0.2
G1 X10.0 Y-0.0 Z0.0 E0.5
G1 X6.2349 Y7.81831 Z0.0 E0.5
G1 X-2.22521 Y9.74928 Z0.0 E0.5
G1 X-9.00969 Y4.33884 Z0.0 E0.5
G1 X-9.00969 Y-4.33884 Z0.0 E0.5
G1 X-2.22521 Y-9.74928 Z0.0 E0.5
G1 X6.2349 Y-7.81831 Z0.0 E0.5
G1 Z0.2
G1 X10.0 Y-0.0 Z0.0 E0.5
G1 X6.2349 Y7.81831 Z0.0 E0.5
G1 X-2.22521 Y9.74928 Z0.0 E0.5
G1 X-9.00969 Y4.33884 Z0.0 E0.5
G1 X-9.00969 Y-4.33884 Z0.0 E0.5
G1 X-2.22521 Y-9.74928 Z0.0 E0.5
G1 X6.2349 Y-7.81831 Z0.0 E0.5
G1 Z0.2
```

Open G-Code file in Cura



Print!

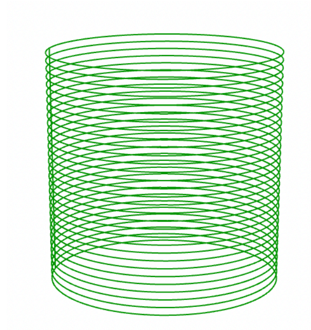


Why generate your own G-Code?

DIRECT MACHINE CONTROL

```
elif (printer=="Eazao" or printer=="eazao"):
    if(self.out_file):
        self.initseq_filename = os.path.join
    self.nozzle = 1.5
    self.extrude_width = 2.2
    self.layer_height = 1.0
    self.extrude_rate = 1.0 #mm extruded/mm
    self.speed = 1500 #mm/minute
    self.printer = "eazao"
    self.resolution = 1.0
    self.x_size = 150
    self.y_size = 150
```

→

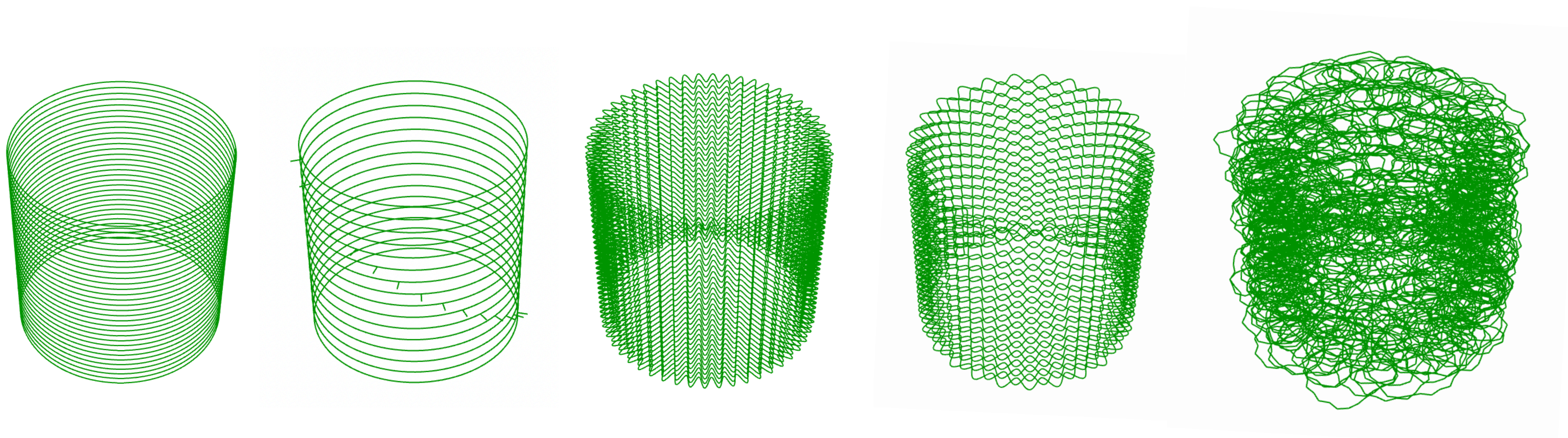


→

- Skip CAD design and slicing software
- Design a toolpath directly by writing code
- Output = 3D printer tool path
- Toolpath determines geometry
- Toolpath also determines surface properties
- Fine-grained control over printer behavior

OUR SOFTWARE LIBRARY

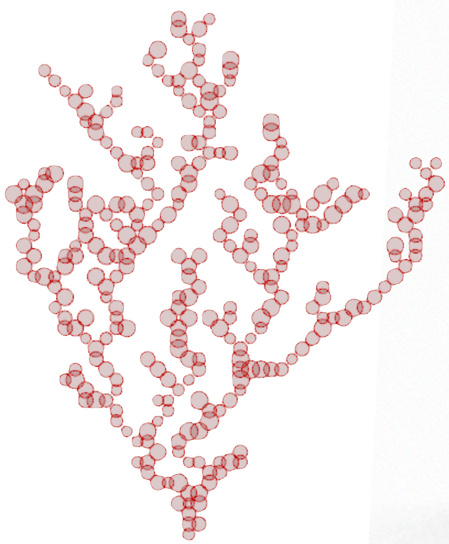
- Slice shape or build directly with toolpath
- Toolpath = slice of object + path pattern



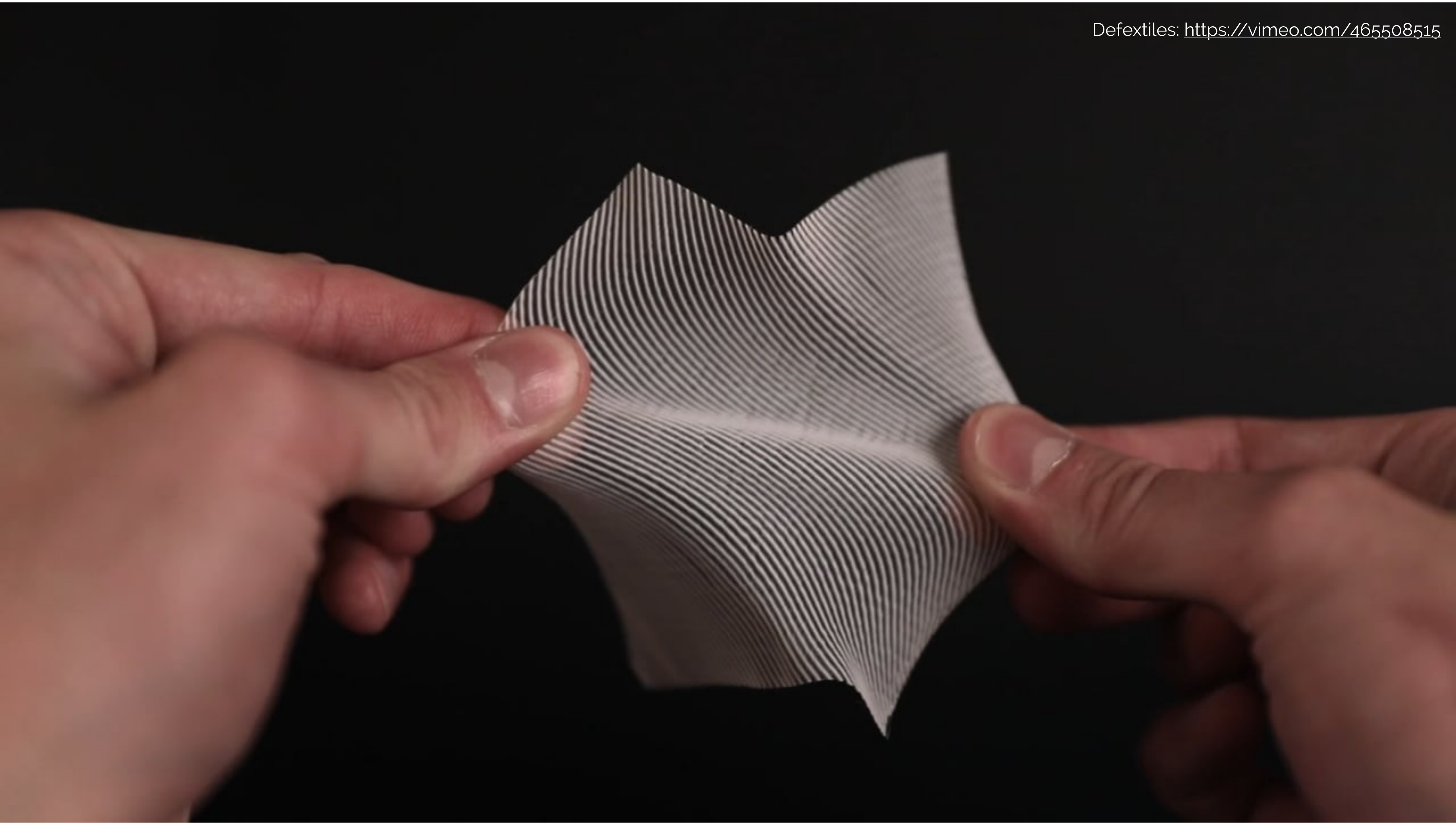
EXAMPLES



ANOTHER EXAMPLE:
2D MATRIX TO VESSEL



More examples


Defextiles: https://vimeo.com/465508515

Artist: LIA

https://www.liaworks.com/
https://www.liaworks.com/theprojects/filament-sculptures/

https://vimeo.com/85913081?embedded=true&source=vimeo_logo&owner=392793

LIA

Modifying Existing G-Code Files

Experiments by Franklin Pezutti-Dyer

“Pug buddy” test print

Alteration of the “Pug buddy” test print in which more and more randomness is added for layers with higher Z-values

Same premise, but with less randomness, and a more gradual increase in randomness


Another example, in which randomness is only added to the right side
(Only perturb coordinates with an X-value above the average X-value)

A different transformation: “twist” the print by rotating X and Y coordinates about a vertical axis, increasing the rotation amount for layers with greater Z-values

Some failed experiments:

NAILED IT!

Thank you!

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