Computational Fabrication

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

Weekly Researcher: Lining Yao

https://morphingmatter.org/

Calendar check in:

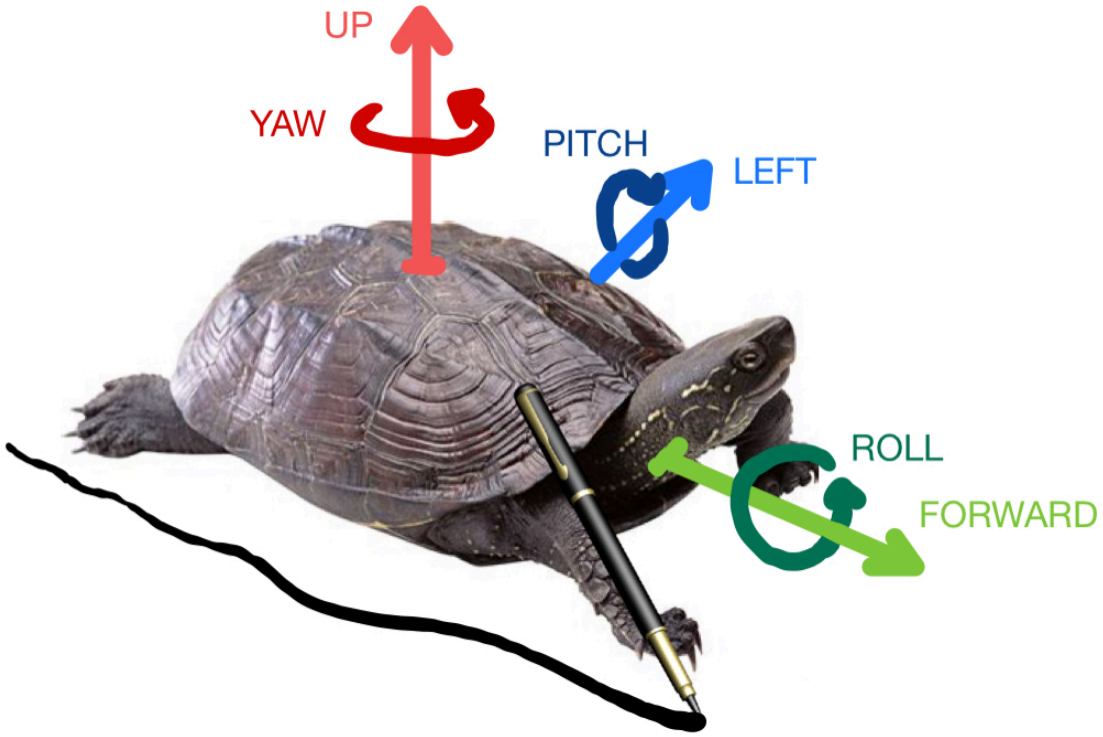
Data Physicalization Assignment

GCODE Assignment

Final Project Proposal Assignment

Hand and Machine
Turtle Library for Generating GCODE

Extruder Turtle Library



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

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

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=1000 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 (moves the 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?

Using the Library

Download or clone the library:
https://github.com/Hand-and-Machine/extruder-turtle-Rhino

Save it with your other class material.
Make a note of the location

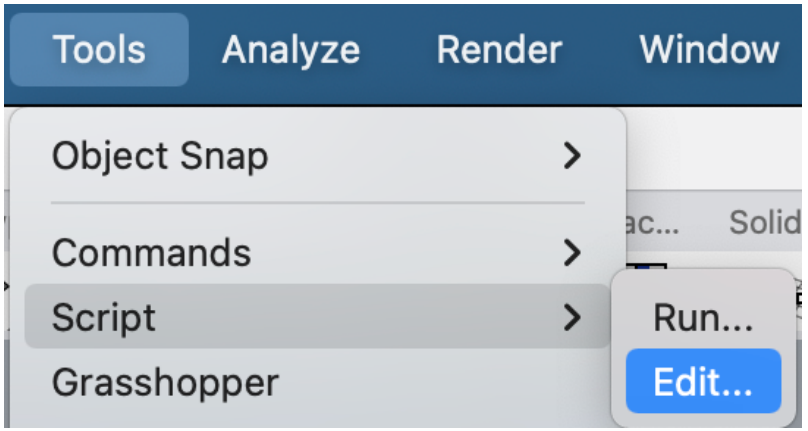
Add the folder where you saved the file to your Rhino path.
We'll go through these steps.

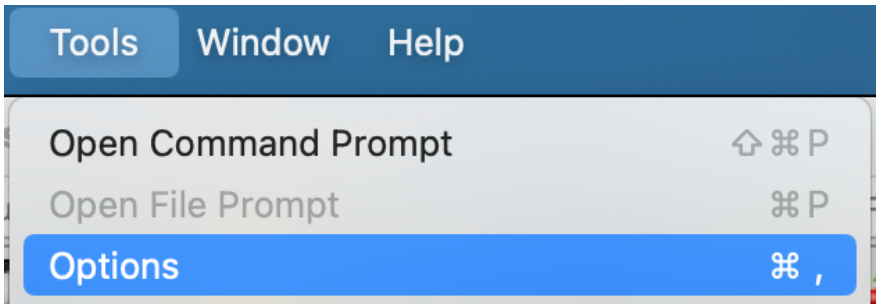
Download again and/or fetch to get most recent version

Add folder to Rhino path

In Rhino:
Go to **Tools** menu.
Select **Script—> Edit**

Then, go to (new) **Tools** menu
Select **Options**



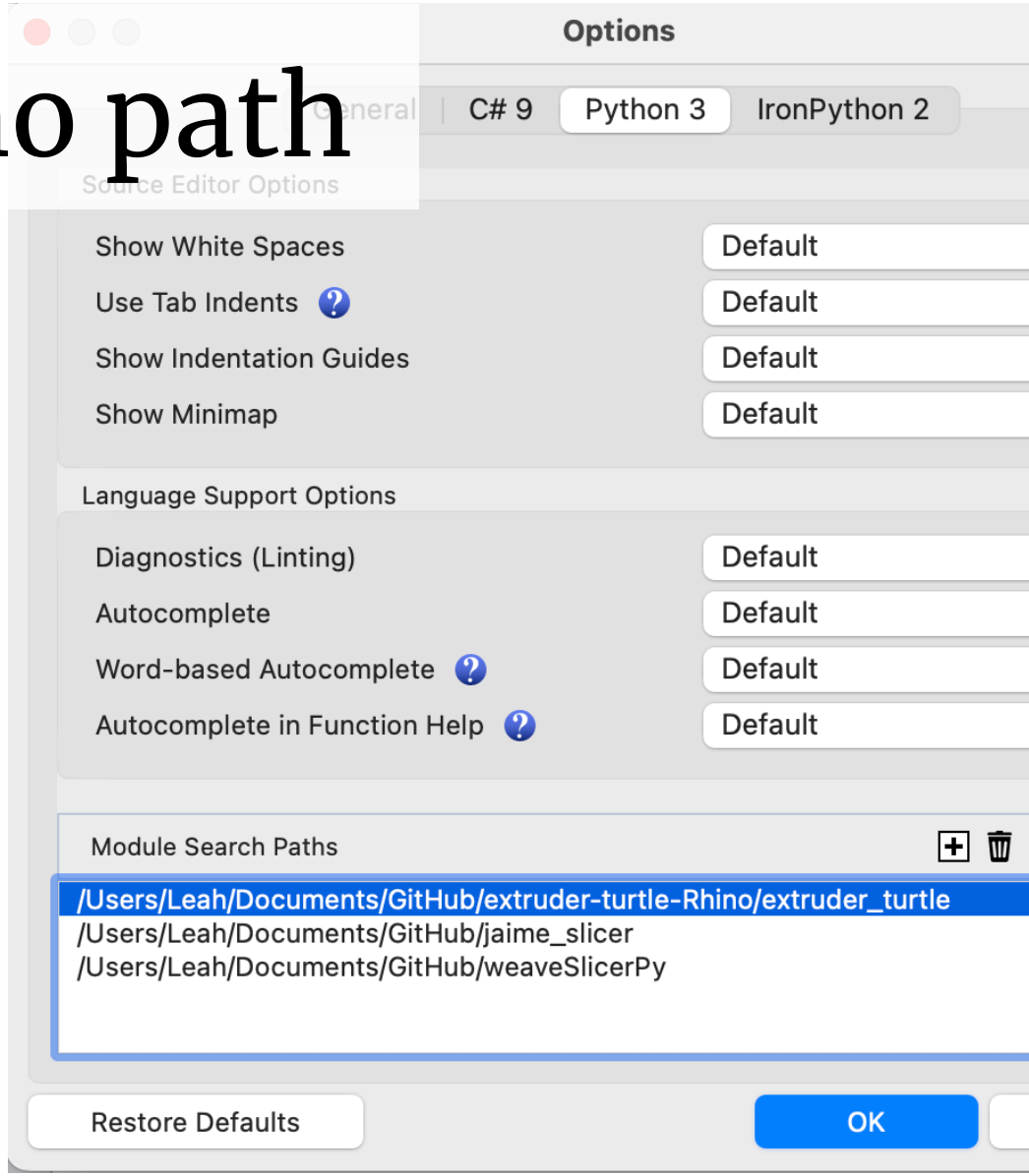


Add folder to Rhino path

In the Options window:
Click on the **Python 3** tab

At the bottom of the window, add the **extruder_turtle** folder to **Module Search Paths**. Note, this folder is inside the top level extruder-turtle-Rhino folder.

Click **OK**



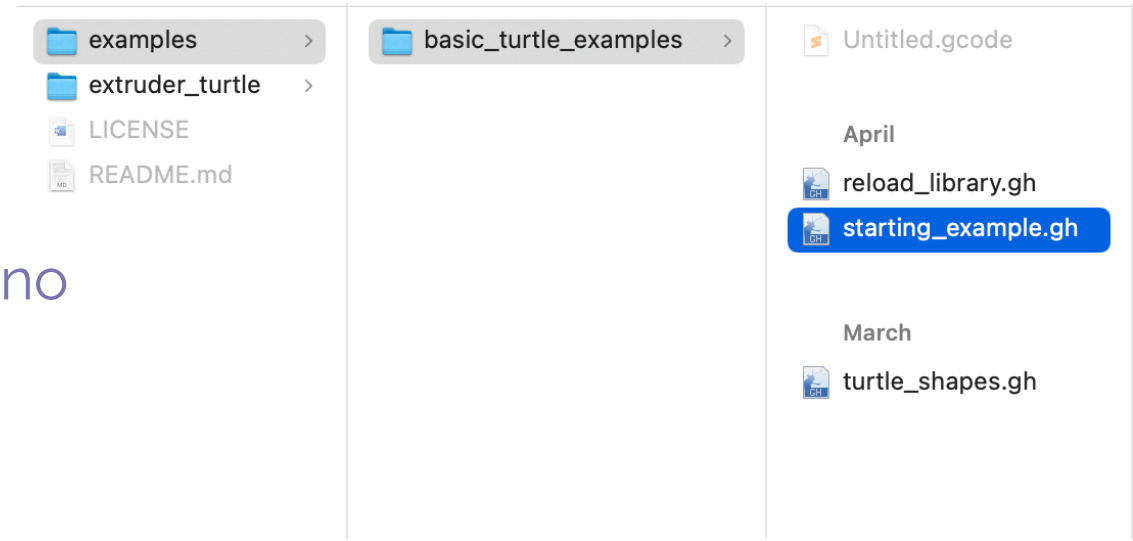
Quit and Restart Rhino

Open Grasshopper

Open an example program

- Navigate to the **examples—>basic_examples** folder in the extruder-turtle-Rhino folder you downloaded

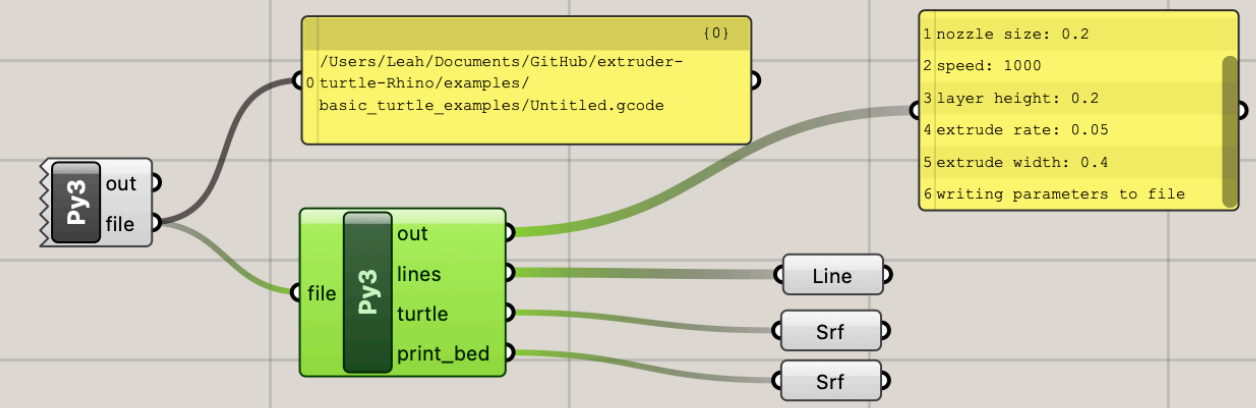
- Open **starting_example.gh**



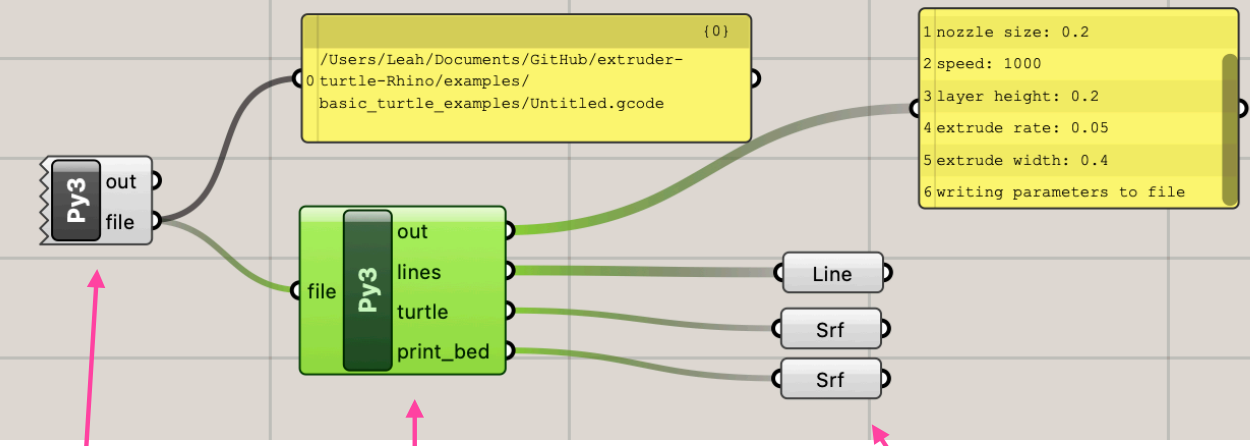
Check to make sure it runs and you don't get library errors. If it doesn't run, something went wrong with installation. Go through the steps again.

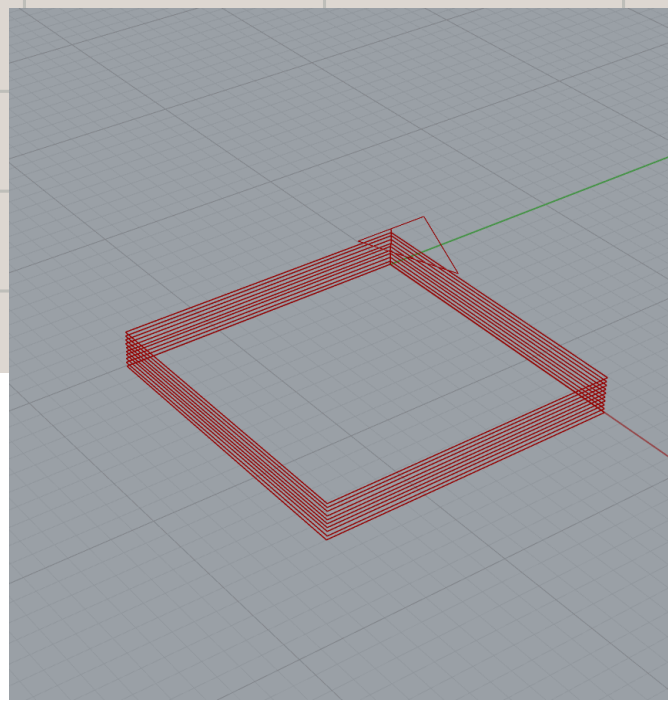
starting_example.gh

This is a simple program that demonstrates how to get started with the Extruder Turtle Library
It generates a gcode file that follows the path of the turtle.
More information: https://handandmachine.org/projects/extruder_turtle_rhino/examples.html



This is a simple program that demonstrates how to get started with the Extruder Turtle Library
It generates a gcode file that follows the path of the turtle.
More information: https://handandmachine.org/projects/extruder_turtle_rhino/examples.html





generates empty .gcode file

generates turtle path and corresponding .gcode commands

3 outputs:
lines: turtle path (.gcode visualization)
turtle: turtle visualization
print_bed: print bed visualization

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

# set up the turtle and associate the
# turtle with a specific printer
t = ExtruderTurtle()
t.setup(filename=file, printer = "ender")

# print out important printer parameters
t.print_printer_information()

# do your turtle programming here
# this code draws a square prism
for j in range (10):
    for i in range (4):
        t.forward(50) # units are in mm
        t.right(90)
    t.lift(t.get_layer_height())
```

Code, top

```
# visualize the turtle's location
# and position
turtle = t.draw_turtle()

# visualize the printer's print bed
print_bed = t.draw_print_bed()

# get the path the turtle has
# traveled to visualize in Rhino
lines = t.get_lines()

# get the approximate print time
t.get_print_time()

# close the generated .gcode file
t.finish()
```

Code, bottom

Run and look at .gcode file

```
G1 F1000
; ********** End printer initialization ***********
; ************************************************

; ***************** Print parameters **************
; Nozzle size: 0.2
; Extrude width: 0.4
; Layer height: 0.2
; Extrude rate: 0.05
; Speed: 1000
; Mix Factor: 0.9
; ************************************************

G1 X50.0 Y0.0 E2.5
G1 X0.0 Y-50.0 E2.5
G1 X-50.0 Y-0.0 E2.5
G1 X-0.0 Y50.0 E2.5
; new layer
G1 Z0.2
G1 X50.0 Y0.0 E2.5
G1 X0.0 Y-50.0 E2.5
G1 X-50.0 Y-0.0 E2.5
G1 X-0.0 Y50.0 E2.5
; new layer
```

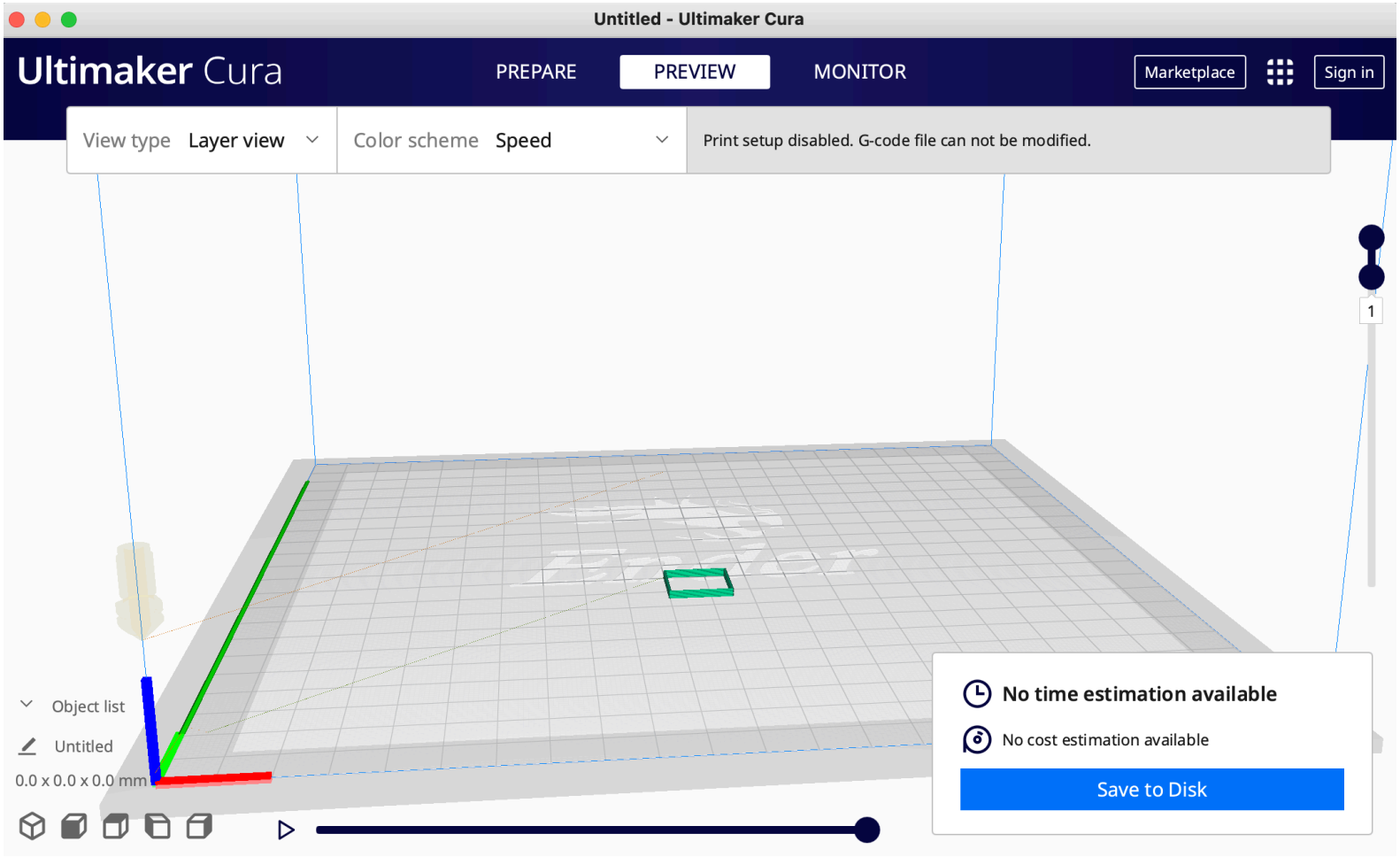
Look at:

Header code
Note: library uses relative coordinates

Move and extrude commands

Layers

Open .gcode file in Cura



Look at library documentation

Extruder Turtle Library

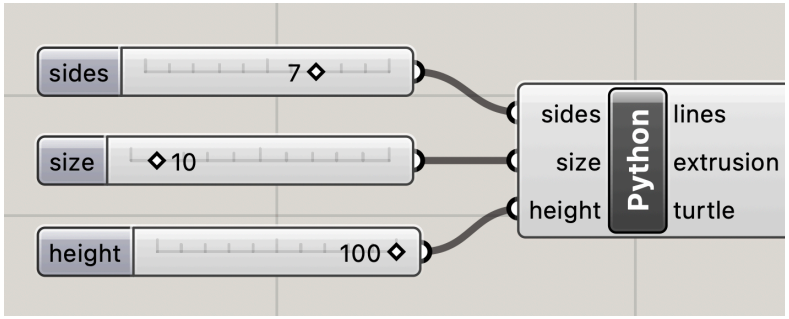
Reference

This page provides detailed documentation on the functions available in the Extruder Turtle Library. Jump to a section:

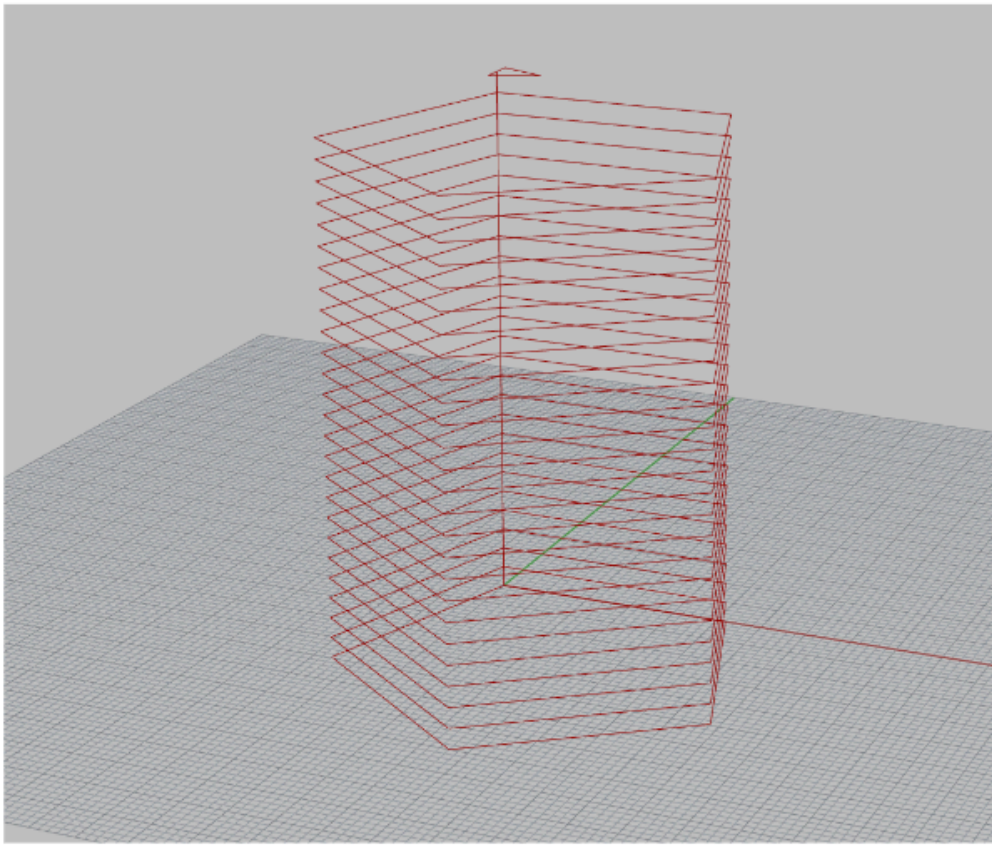
Setup	
`ExtruderTurtle`	Constructor. Generates a turtle object. **Parameters**: N/A **Returns:** A turtle object located at the origin. **Example:** `t = ExtruderTurtle()`
`setup`	Sets initial parameters for the turtle. A printer must be set either through this function or the set_printer function for the printer-based functions to work. A filename must be specified via this function for g-code generation to work. The library currently supports the following printers: Eazao Zero, "eazao" 3D Potter Super 10, "super"

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

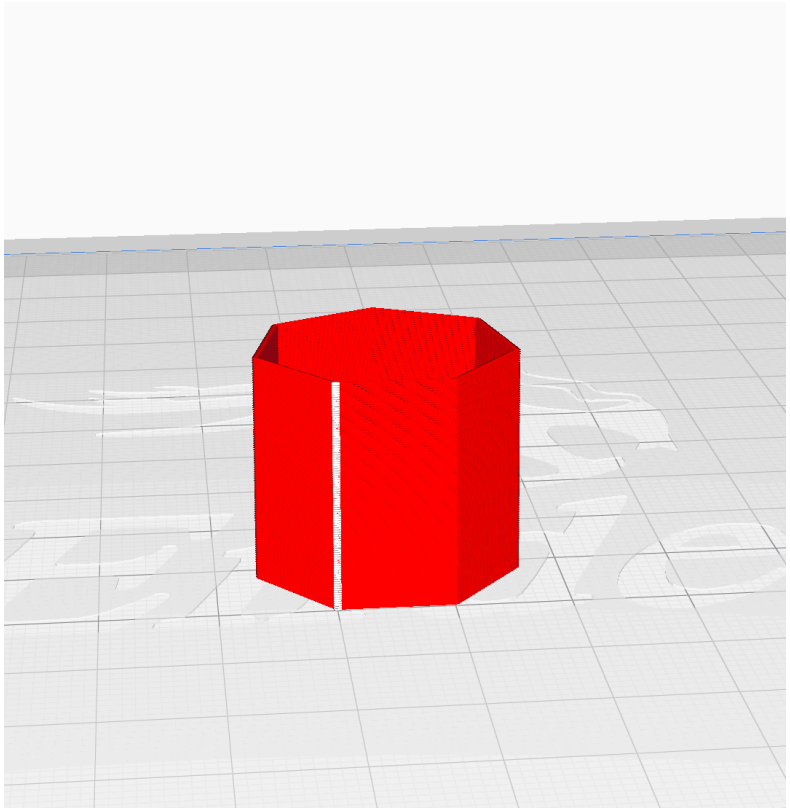
Edit your code to make polygon prisms



new inputs:
sides: number of sides
size: size of sides
height: height of prism



Open G-Code file in Cura



Printer settings and the library code

Open reload_library.gh

- Navigate to the **examples—>basic_examples** folder in the extruder-turtle-Rhino folder you downloaded

- Open **reload_library.gh**

Open reload__library.gh

```
# code required so that library reloads
# use if you want to make edits to library
# and have the changes compiled here
# must be above other import statements
import sys
import operator as op
to_delete = list()
for module in sys.modules:
    if "extruder_turtle" in module:
      to_delete.append(module)
for module in to_delete:
    sys.modules.pop(module)

# begin standard import statements
import rhinoscriptsyntax as rs
import extruder_turtle
from extruder_turtle import *
```

deletes previous library import and reloads each time this block compiles

Note: this basic framework allows you to use external python files in GH programs

Look at library python files

Open ExtruderTurtle.py

- Navigate to the **extruder_turtle** folder in the extruder-turtle-Rhino folder you downloaded

- Open **ExtruderTurtle.py** in a text editor of your choice

Open ExtruderTurtle.py

```
import os
import math
import copy
import rhinoscriptsyntax as rs
__location__ = os.path.dirname(__file__)


class ExtruderTurtle:

    def __init__(self):
        self.x = 0
        self.y = 0
        self.z = 0
        self.forward_vec = [1, 0, 0]
        self.left_vec = [0, 1, 0]
        self.up_vec = [0, 0, 1]
        self.use_degrees = True
        self.pen = True
        self.mix_factor = 0.9

        # GCODE writing and history tracking
        self.write_gcode = False
        self.track_history = True
        self.prev_points = [(self.x,self.y,self.z)]
        self.line_segs = []
        self.extrusion_history = []
```

set_printer function

```
# set printer parameters
def set_printer(self,printer):
    if (printer=="Ender" or printer=="ender" or printer=="creatlity" ):
        if(self.out_file):
            self.initseq_filename = os.path.join(__location__, "data/initseqEnder.gcode")
        self.nozzle = 0.2
        self.extrude_width = 0.4
        self.layer_height = .2
        self.extrude_rate = 0.05 #mm extruded/mm
        self.speed = 1000 #mm/minute
        self.printer = "ender"
        self.resolution = .1
        self.x_size = 220
        self.y_size = 220
    elif (printer=="super" or printer=="3Dpotter" or printer=="3D Potter"  or printer=="3d pot
        if(self.out_file):
            self.initseq_filename = os.path.join(__location__, "data/initseq3DPotter.gcode")
        self.nozzle = 3.0
        self.extrude_width = 3.4 #mostly for solid bottoms
        self.layer_height = 2.2
        self.extrude_rate = 3.0 #mm extruded/mm
        self.speed = 1000 #mm/minute = 16.6 mm/second
        self.printer = "super"
```

edit to add/represent your printer

```
# set printer parameters
def set_printer(self,printer):
    if (printer=="Ender" or printer=="ender" or printer=="creatlity" ):
        if(self.out_file):
            self.initseq_filename = os.path.join(__location__, "data/initseqEnder.gcode")
        self.nozzle = 0.2
        self.extrude_width = 0.4
        self.layer_height = .2
        self.extrude_rate = 0.05 #mm extruded/mm
        self.speed = 1000 #mm/minute
        self.printer = "ender"
        self.resolution = .1
        self.x_size = 220
        self.y_size = 220
    elif (printer=="super" or printer=="3Dpotter" or printer=="3D Potter"  or printer=="3d pot
        if(self.out_file):
            self.initseq_filename = os.path.join(__location__, "data/initseq3DPotter.gcode")
        self.nozzle = 3.0
        self.extrude_width = 3.4 #mostly for solid bottoms
        self.layer_height = 2.2
        self.extrude_rate = 3.0 #mm extruded/mm
        self.speed = 1000 #mm/minute = 16.6 mm/second
        self.printer = "super"
```

edit to add/represent your printer

```
# set printer parameters
def set_printer(self,printer):
    if (printer=="Ender" or printer=="ender" or printer=="creatlity" ):   name
        if(self.out_file):
            self.initseq_filename = os.path.join(__location__, "data/initseqEnder.gcode")
        self.nozzle = 0.2
        self.extrude_width = 0.4
        self.layer_height = .2
        self.extrude_rate = 0.05 #mm extruded/mm
        self.speed = 1000 #mm/minute
        self.printer = "ender"
        self.resolution = .1
        self.x_size = 220
        self.y_size = 220
                                  bed size
    elif (printer=="super" or printer=="3Dpotter" or printer=="3D Potter"  or printer=="3d pot
        if(self.out_file):
            self.initseq_filename = os.path.join(__location__, "data/initseq3DPotter.gcode")
        self.nozzle = 3.0
        self.extrude_width = 3.4 #mostly for solid bottoms
        self.layer_height = 2.2
        self.extrude_rate = 3.0 #mm extruded/mm
        self.speed = 1000 #mm/minute = 16.6 mm/second
        self.printer = "super"
```

questions?

Turtle Geometry & Euclidean Geometry are friends

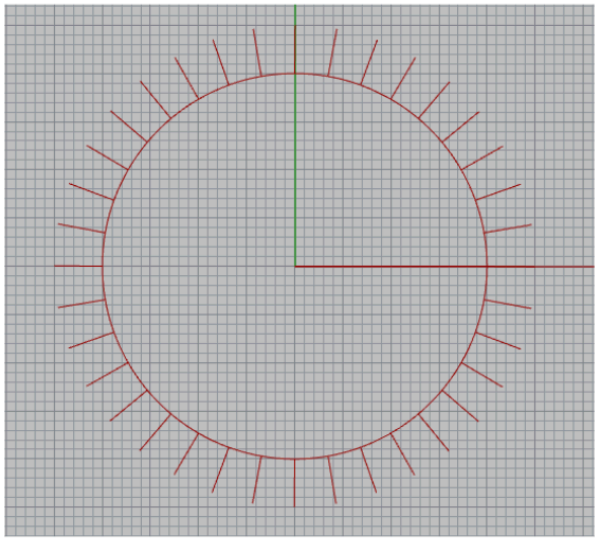
Open a new example program

- Navigate to the **examples—>basic_examples** folder in the extruder-turtle-Rhino folder you downloaded

- Open **euclidean_geometry.gh**

euclidean_geometry.gh

```
for i in range (0,361):
    # euclidean geometry: circle
    x = radius * math.cos(math.radians(i))
    y = radius * math.sin(math.radians(i))
    # set the turtle's position using coordinates
    t.set_position(x,y)
    if (i>0 and i%10 == 0):
        # turtle geometry: spikes
        t.right(90)
        t.forward(5)
        t.back(5)
        t.left(90)
```



Look at library documentation

Extruder Turtle Library

Reference

This page provides detailed documentation on the functions available in the Extruder Turtle Library. Jump to a section:

Setup	
`ExtruderTurtle`	Constructor. Generates a turtle object. **Parameters**: N/A **Returns:** A turtle object located at the origin. **Example:** `t = ExtruderTurtle()`
`setup`	Sets initial parameters for the turtle. A printer must be set either through this function or the set_printer function for the printer-based functions to work. A filename must be specified via this function for g-code generation to work. The library currently supports the following printers: Eazao Zero, "eazao" 3D Potter Super 10, "super"

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

Large Assignment 4: GCODE

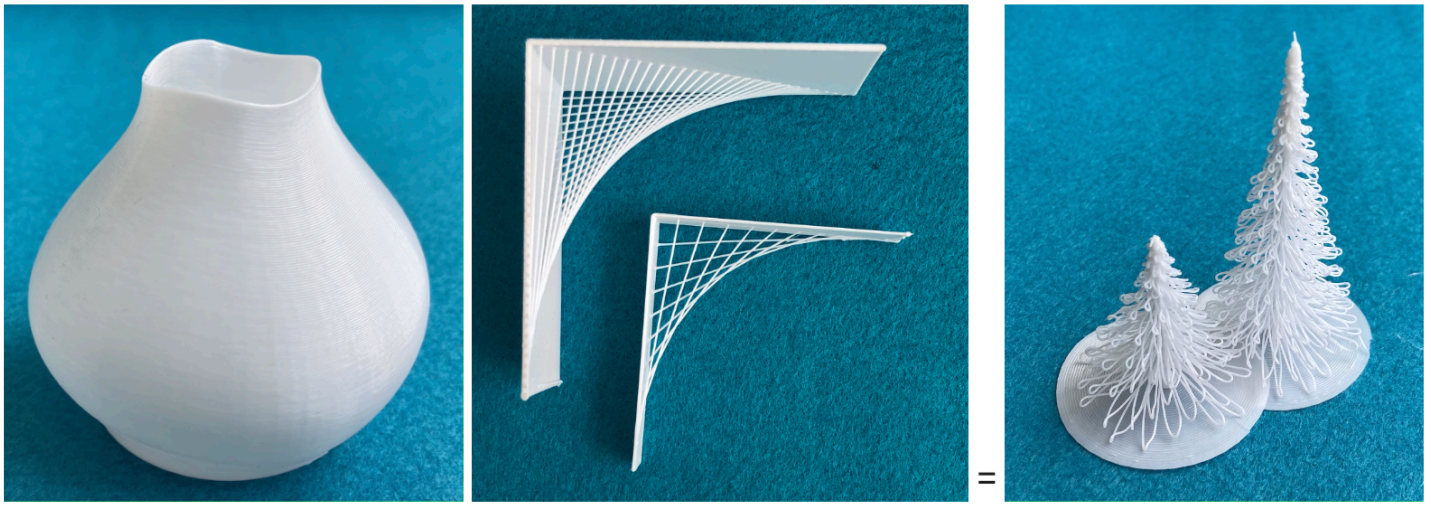
Due Dates

Assignment : End of day, Tuesday October 22

Presentation: Thursday October 24

Initial comments: End of day, Thursday October 24

Comment responses: End of day, Friday October 25



Option: use clay 3D printer

Thank you!

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