Final Project

What is this document?

- Final Project Brief
- A template document for your team project. (If you work independently, you still need to document your progress in this document.)
- You use this document to manage and keep track of your teamwork progress.
- Every time we meet about your project, you must bring this document and keep track of notes, discussions, tasks, and plans.
- You should submit a copy of this document organized as a design progress booklet at the end of the semester.

Project Progress

Project	Due	Status	Notes
Finalize Proposal of ideas & concepts	Nov 12, 2024	Finished	Ideation: Image sampler + Drawing input + Cellular Automata output
Checkpoint (Nov 28th)	Nov 28, 2024	Finished	Feasibility of ideas
Final Presentation (Dec. 5th)	Nov 5, 2024	Finished	UI -Human, Remote Control Panel
Submission of Final Deliverable (Dec. 12th)	Dec 12, 2024	Finished	Drawing constraint outside the boundary Color, polygon test

Final Project Brief

This course prepares students for modeling geometry through the scripted development of parametric schemes primarily for design applications. To introduce students to basic scripting in a geometrical modeling environment focusing on form-making algorithms and reinforce and extend basic concepts of parametric modeling.

During this semester, the course covered geometric concepts, including points, vectors, curves, surfaces, B-Rep, and meshes. It then shifted focus to practical applications by reproducing classical form-finding algorithms such as L-System, Cellular Automata, and Boid Simulation.

The final project challenges students to design a small platform or tool featuring a unique user interface within the Rhino Grasshopper environment. This can take the form of a practical application or an engaging interactive game. The most important requirement is that it must be primarily built using Python in the Rhino and Grasshopper environment.

Recommend plug-in for UI development:

1. **Human + Human UI**
2. **LINK_Dashboards**
3. **ArchiVision**

Cover Page

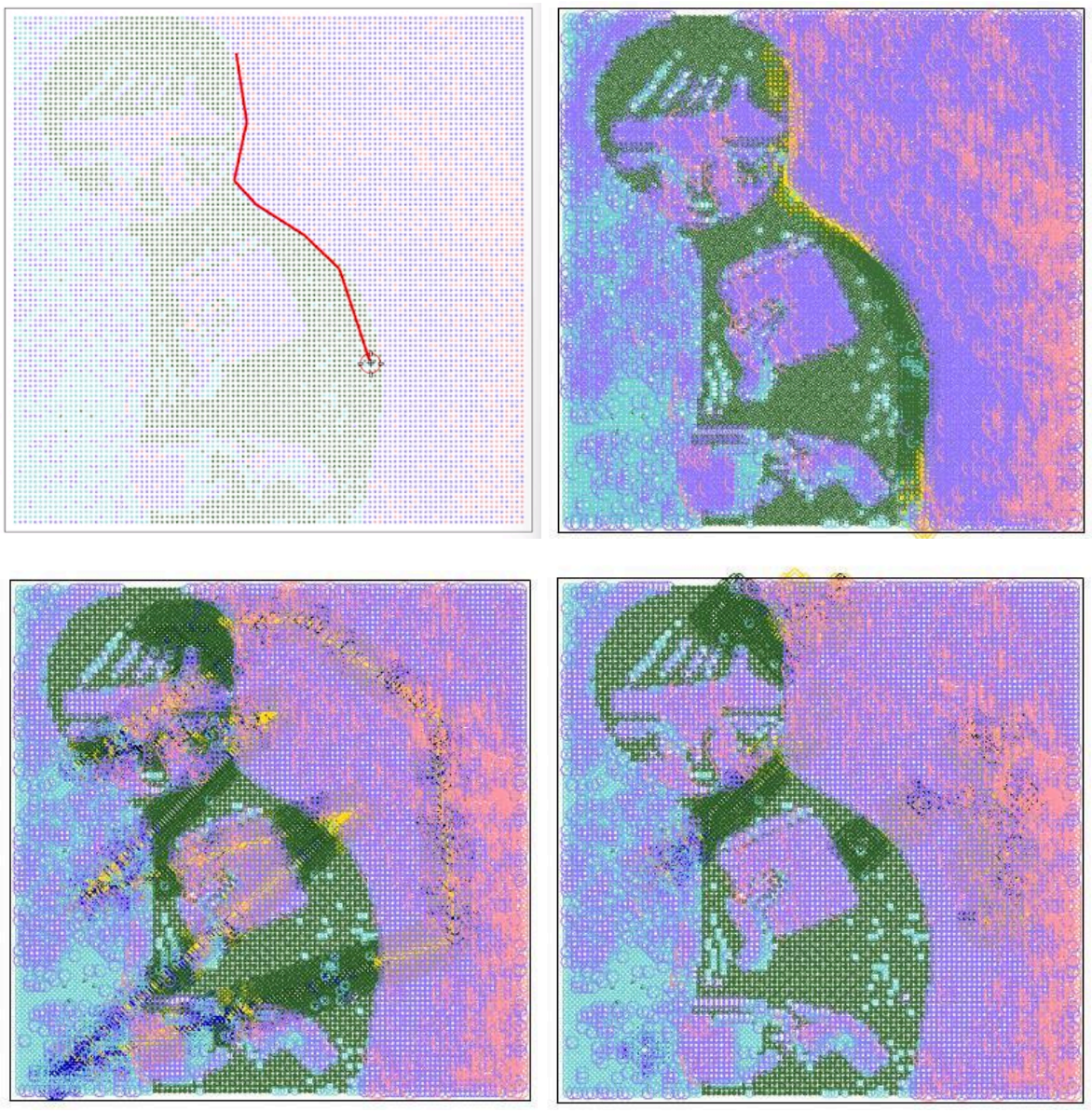


Fig.1 Life of Drawing

Life of Drawing

Cellular Automata interactive drawing tools in Rhinoceros with Grasshopper

Project Introduction
Team Members: Sohyun Jin
Role & Contribution: Case study, ideation, coding, documentation, presentation

Week One: Ideation & Final Proposal

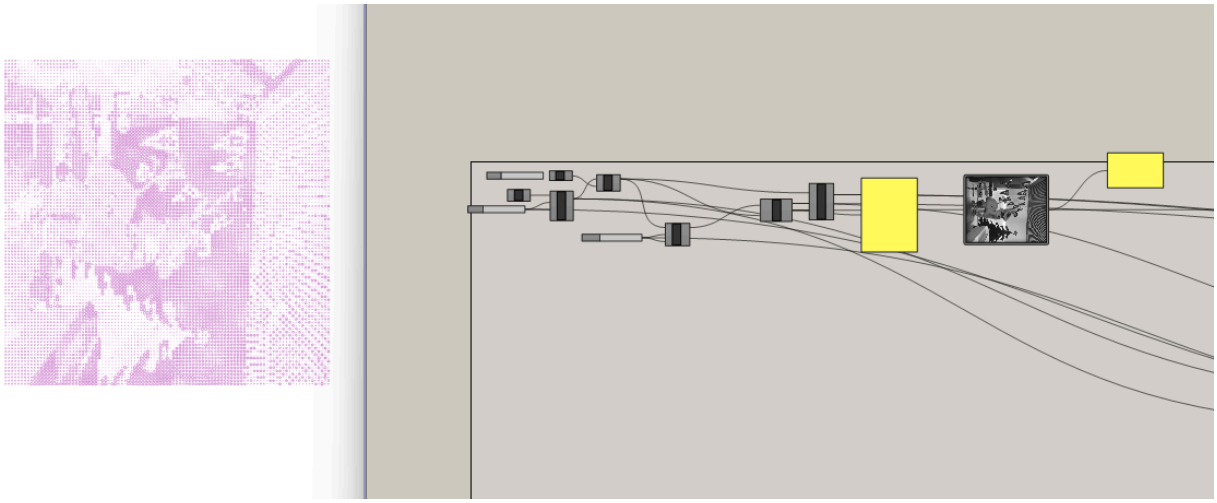


Figure 1. Image sampler test

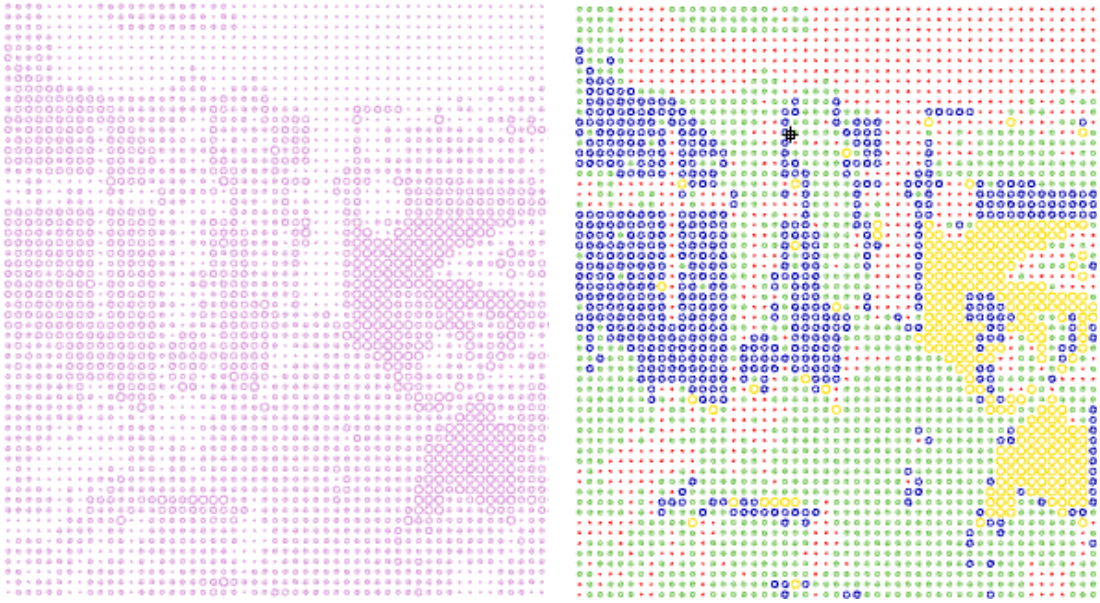


Figure2. Circle size according to brightness (left) Color types according to four domains in brightness (right)

Nov 12, 2024 |

Progress: Discussion

Notes

Jimmy suggested I make a drawing tool via cellular automata, and I felt intrigued. I wanted to employ OpenCV to deal with image data, thinking I could incorporate it into the cellular automata. Instead of using Kernel, I took advantage of grasshopper functionalities. They have a well known component named Image sampler, although I have had no chance to use it. This week I have explored two and seen the possibilities: image sampler and free-drawing methods in Grasshopper.

Action items

- ☑ ~~Case study~~
- ☑ ~~Explore image sampler on Grasshopper~~
- ☑ ~~Explore other tools~~

Nov 14, 2024 |

Progress: Finalize Project Proposal

Notes

What I have learned: The Image sampler component requires pre-reparameterization of the coordinates to convert the size of points in the rectangle within the (0, 0, 0) and (1, 1, 1). Visited RhinoDeveloper site, accessing Rhino API reference, learning there has been well arranged C# documentation for Rhinoceros. This project felt like exploring the possibilities restricted in Rhinoceros 3D.

Action items

- ☑ ~~Explore image sampler on Grasshopper~~
- ☑ ~~Explore C# Rhino Developer documentation~~

Week Two: Project Development

```
Command: Grasshopper
let's check0
let's check0
Mouse Down Enabled
let's check1
Mouse Down Enabled
let's check2
Mouse Down Enabled
let's check3
Mouse Down Enabled
```

Figure 3. Whenever mouse down, time delay occurs. It was challenging to control the mouse event.

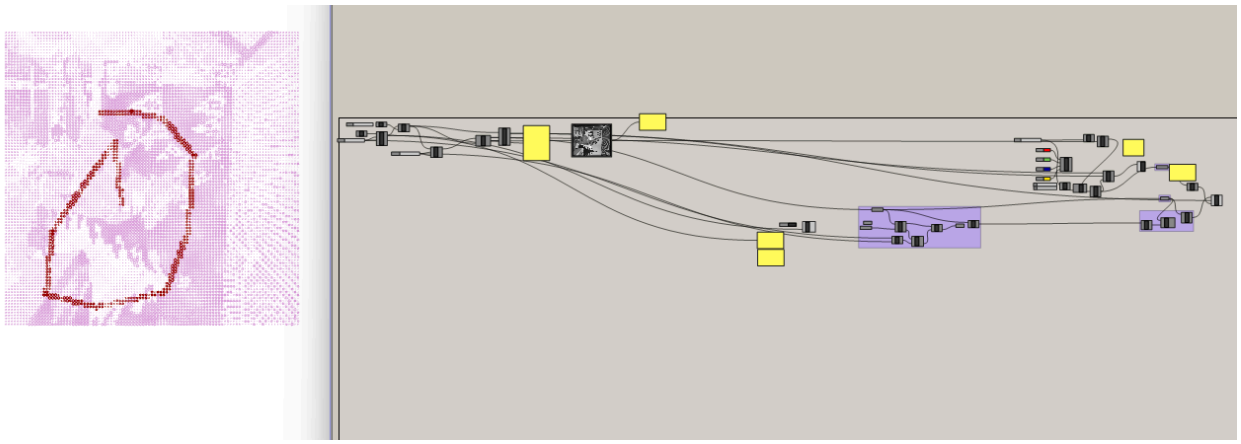


Figure 4. Drawn polylines respond to a circle set of the Image sampler result, marked as red. (Smaller than or equal to grid cell size (1))

Nov 19, 2024 |

Progress Tool development1

Notes + Guest lecture (Nike designer)

Exploring free drawing tools in Rhino Python, RhinoScriptPython, Rhino C#. Learned that David Rutten intentionally applied Mouse Event to Grasshopper as little as possible as he didn't like timer. (Read it here) Also learned importing datatype 2D list to grasshopper is more challenging than I thought. I adjust myself to the given condition, converting coordinates into 1D list for Grasshopper python.

Action items

- ☑ ~~Combine the Image sampler and a Polyline drawing tool~~
- ☑ ~~Explore other possibilities~~

Nov 21, 2024 |

Progress Tool development2

Notes from the Workshop session

Explored `Windows.Forms.Control.MouseButtons`, `Windows.Forms.Cursor.Position`, or `Rhino.UI.MouseCallbackEventArgs` through Rhino C#. However these caused serious delays in Grasshopper processing. Image sampler sometimes caused troubles as well, thus I needed to check the compatibility of components I used. For now, I just adhered to a recursive polyline drawing function that requires the user to click a certain number of points. A minor issue with this method is the user can't see the points until all the required clicks are completed. I will add this to the action items next week. Anyway, it is time to move on to cellular automata.

Action items

- ☑ ~~Find best solutions for drawing tools.~~

Week Three: Project Development

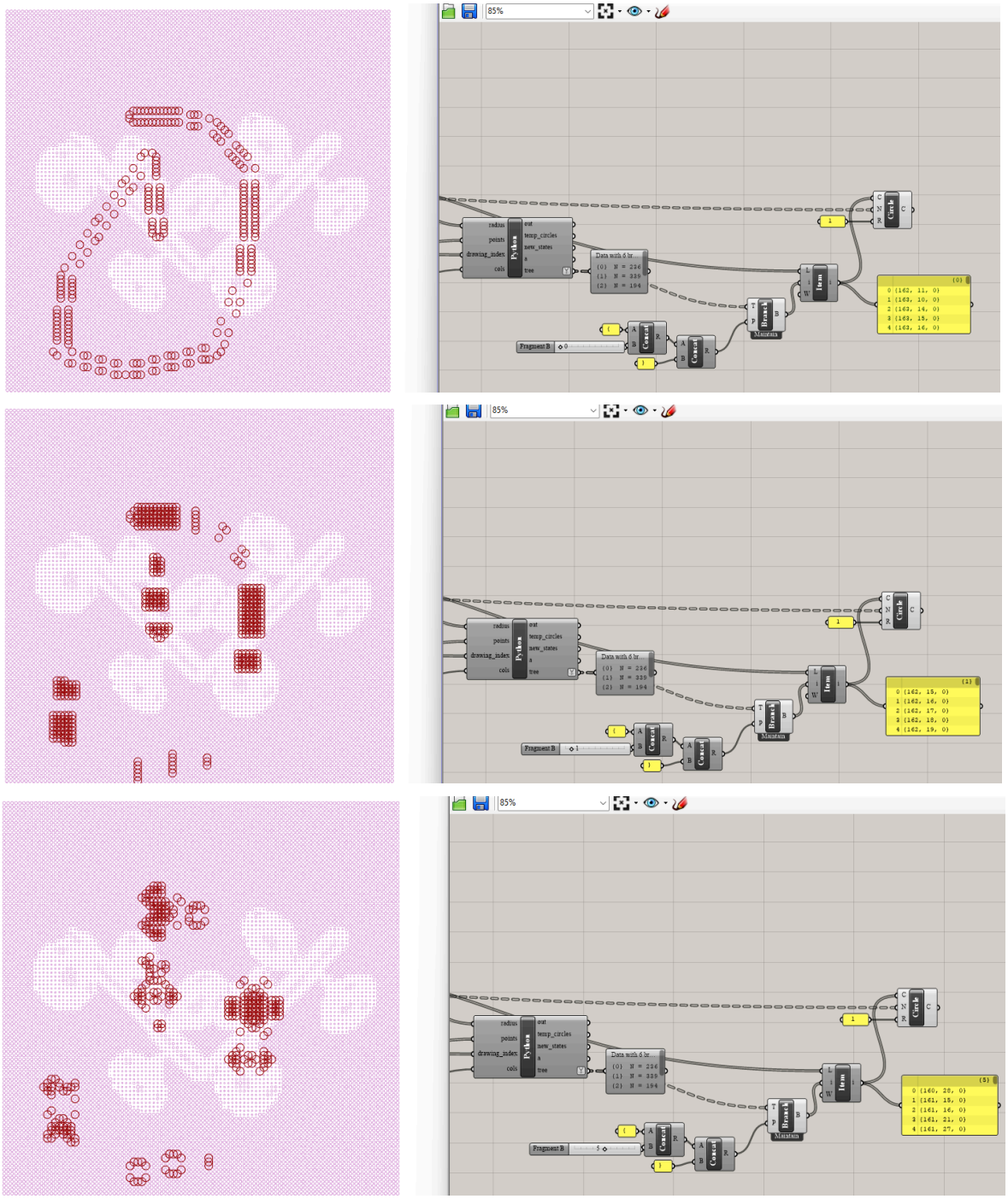
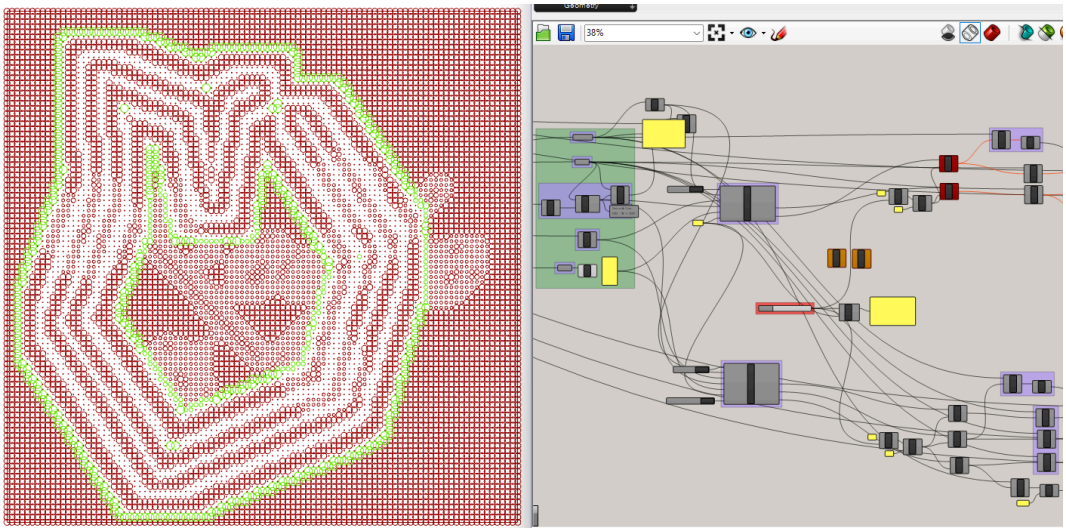


Figure 5.



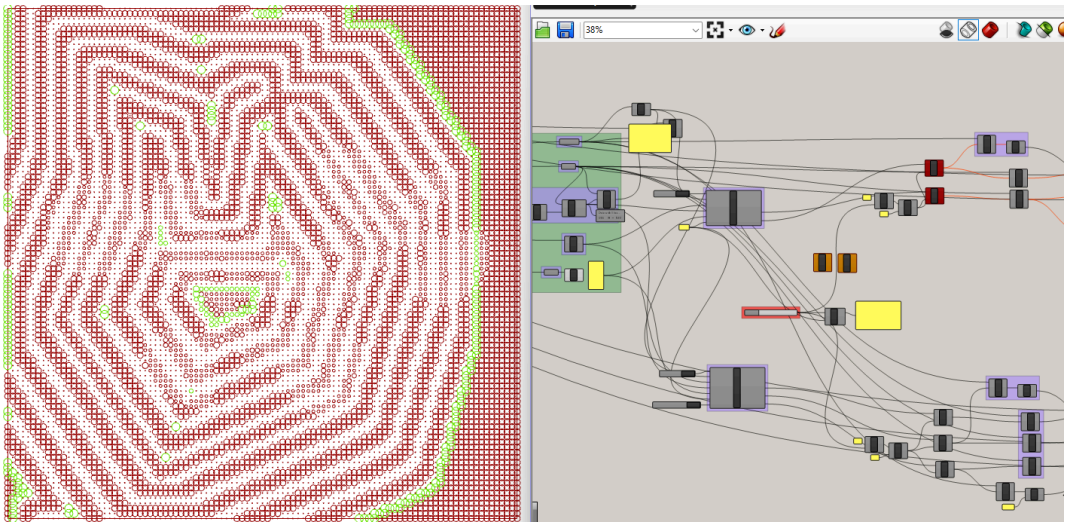


Figure 6.
Figure 7.

Nov 26, 2024 |

Progress

Notes from the Workshop session

- Based on the Cellular automata code during lecture, change it into meaningful outputs which enable it to serve in Grasshopper components. For example, geometry types on the grid points are created by component connections from the sorted lists from GH python.
- Animation effects could not follow the approach from lecture, using rs.EnableRedraw(False). Currently number sliders can point to each phase state. Later, users can save the avi animation file of their personal *drawing of life*.

Action items

- ☑ ~~A minor issue in the polyline drawing method: users can't see the points until all the required clicks are completed.~~
- ☑ ~~Test with Cellular Automata model in Grasshopper~~
- ☑ ~~Test with non-block state~~

Nov 28, 2024 |

Progress

Notes from the Workshop session

- I have tested UI using playground and Human. Using the remote control panel in Grasshopper makes the process slower. It would be better to implement Human UI or other options, like patience.
- Create different modes: Cellular Automata, Ripple effect A, Riffel effect B. The codes work successfully, and all I need to add is color, pixel type, or variations.

Action items

- ☑ ~~How to make a gif effect in grasshopper?~~
- ☑ ~~Converting data into Grasshopper format~~

Week Four: Debug and Review

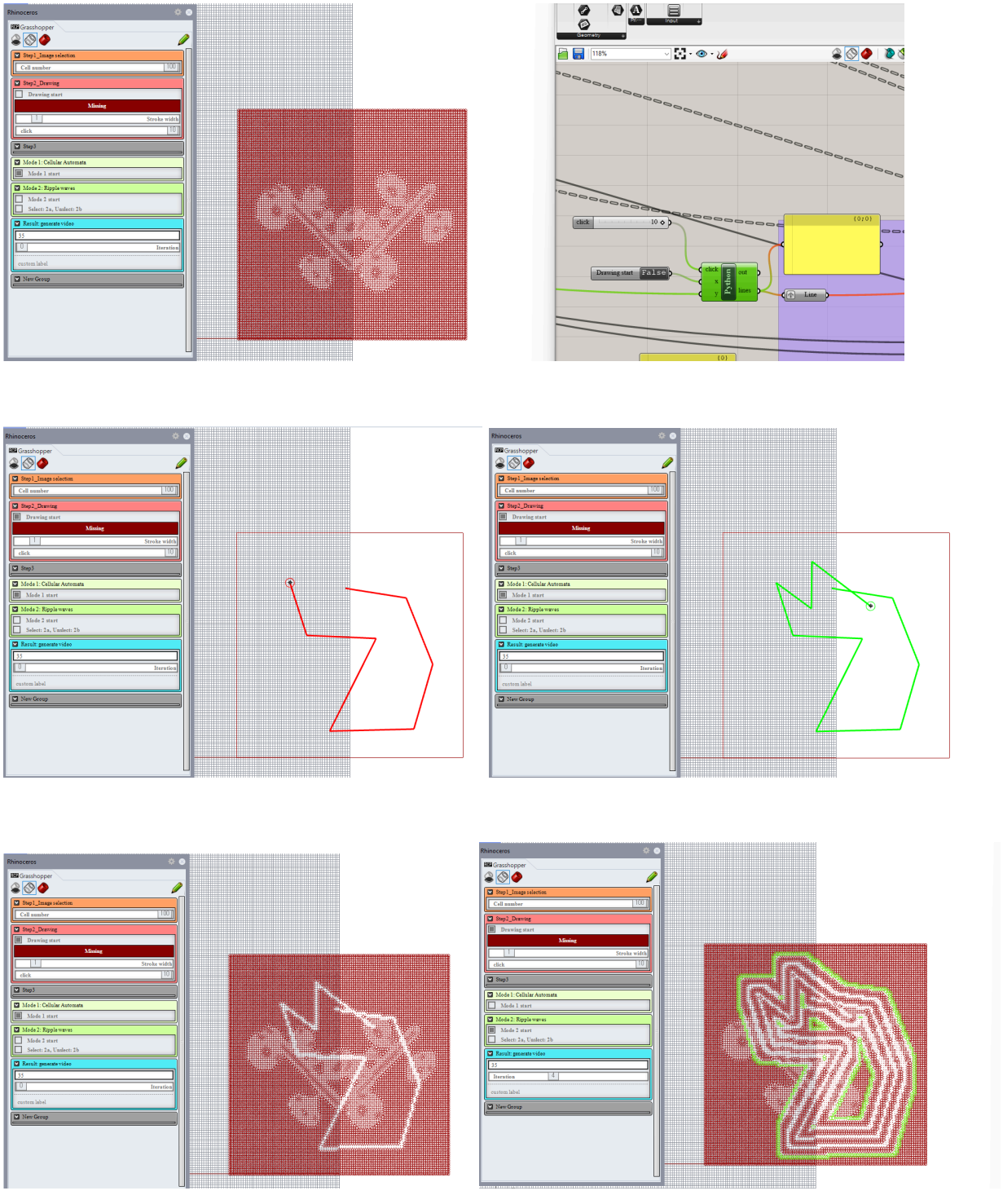


Figure 8.

Dec 3, 2024 |

Progress

Notes from the Workshop session
Test Human, Human UI, Grasshopper remote control panel

Action items

- ☑ ~~Add boundary box for the drawing constraint~~
- ☑ ~~arduino.exe file error remove~~

Dec 5, 2024 |

Progress: Final Review

Notes from the Final Review

Jimmy advised that changing the UI does not resolve the slow performance issue. The use of a remote control panel in Grasshopper exacerbates the delay when shifting parameters.
I decided to focus on visual effects to complete the project.

Action items

- ☑ ~~Explore Human UI, LINK_Dashboards, ArchiVision~~
- ☑ ~~More visual effects for fitting the concept (polygon, circle, colors)~~

Week Five: Final Project Report

Dec 13, 2024 |

Final Submission:

Life of Drawing

Drawing for interactive image in Rhinoceros with Grasshopper
(Keywords: Image sampler, Cellular Automata, Ripple effects, Free-drawing)

1. Introduction

The starting point of this project was when I complained about the existing limitations in Rhinoceros and Grasshopper. Jimmy suggested that I make a drawing tool with a cellular automata effect, as an extension of my previous work with Sherry, One-man drawing game, Exquisite Corpse. It was an intriguing idea, especially the part where users can directly input the position of live cells. While using Image Sampler, the default point group on the image is dead, the closest point group near the user's new drawing line is alive.

Goal: To work within the Rhinoceros and Grasshopper environment and learn how to adapt to this program through explorative practices. I usually use Rhinoceros 3D with accustomed tools, but by pushing my boundaries, I aim for this project to broaden my perspective and enhance my flexibility.

2. Relevant work (Precedents)

There are many interesting projects that utilize the Image Sampler component in Grasshopper. It is a powerful tool for creating visual effects and is often used to transform images into 3D geometry. (Figure 10) However, examples of integrating mouse input directly with the Image Sampler are rare. This may be because other tools, such as p5.js, Processing, or Pillow, offer more straightforward ways to achieve interactive image manipulation outside of Rhino3D.

Figure 9. Generative Landscape, Image Sampler examples
https://generativelandscapes.wordpress.com/2014/08/14/image-sampler-example-2-5/

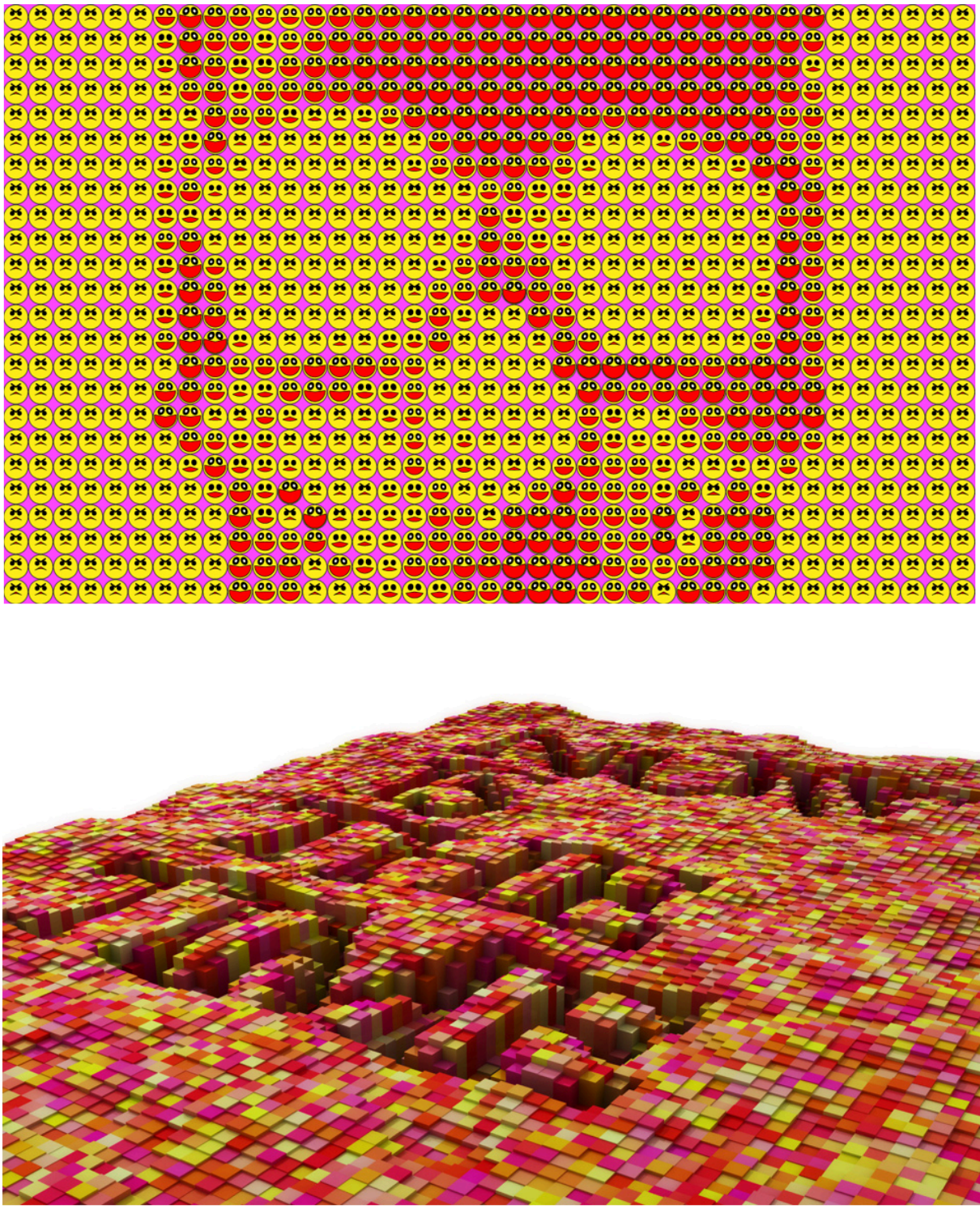


Figure 10. parametric image sampling, John Locke
http://gracefulspoon.com/blog/tag/grasshopper/

3. Method (Platform Design Strategy)

A basic workflow:

1. Using image sampler, map the point coordinates, colors, polygons according to the 1D list data.
2. Default image sampler point group is dead, the closest point group near the user's new drawing line is alive.
3. Dead and live groups will be distinctive. Color/style/shape/size difference
4. Add other modes than just Cellular Automata. Ripple effect.
5. Collect parameters to the Remote Control Panel

4. Result

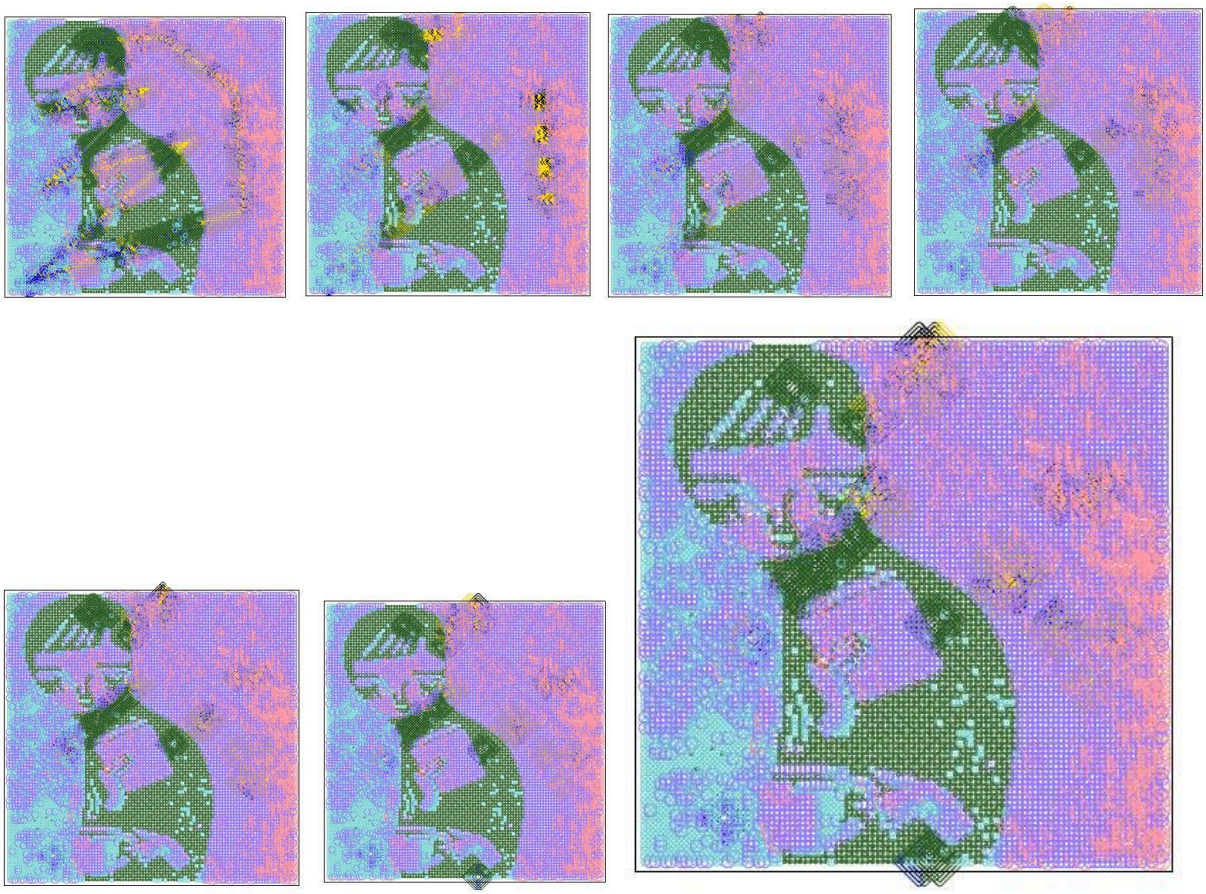


Figure 11. Mode 0: Cellular Automata (cell size 100) GIf

4-1. How does **Mode 0: Cellular Automata** work?

When the number of rows and columns is reduced to 36, it becomes easier to distinguish between the live and dead cells. The dead cells are mapped using color and circle radius based on the intensity of the original colors in the uploaded image. Initially, I intended to use RGB colors directly, but since GH Python doesn't handle RGB data flexibly, I opted to work with the intensity values, dividing the colors into four groups. I think this simplified approach works well.

The live cells for each phase will change shape into a curvy diamond. The color change corresponds to the cell locations; for example, **purple → yellow, green → green, pink → blue, and cyan → black.** I experimented with hatch and surface options to make the cells more distinct, but this proved to be very time-consuming. As a result, I decided to use layered polygons to mark them.

In the actual demonstration (Cell size 100), the dead cells, represented by circles, became dense and effectively visualized the uploaded image. The movement of the live cells was subtle yet clearly observable. I appreciate how the cellular automata effect emerges gently, with bright or darker colors, to present the game of drawing life, as if a city seen from the satellite.

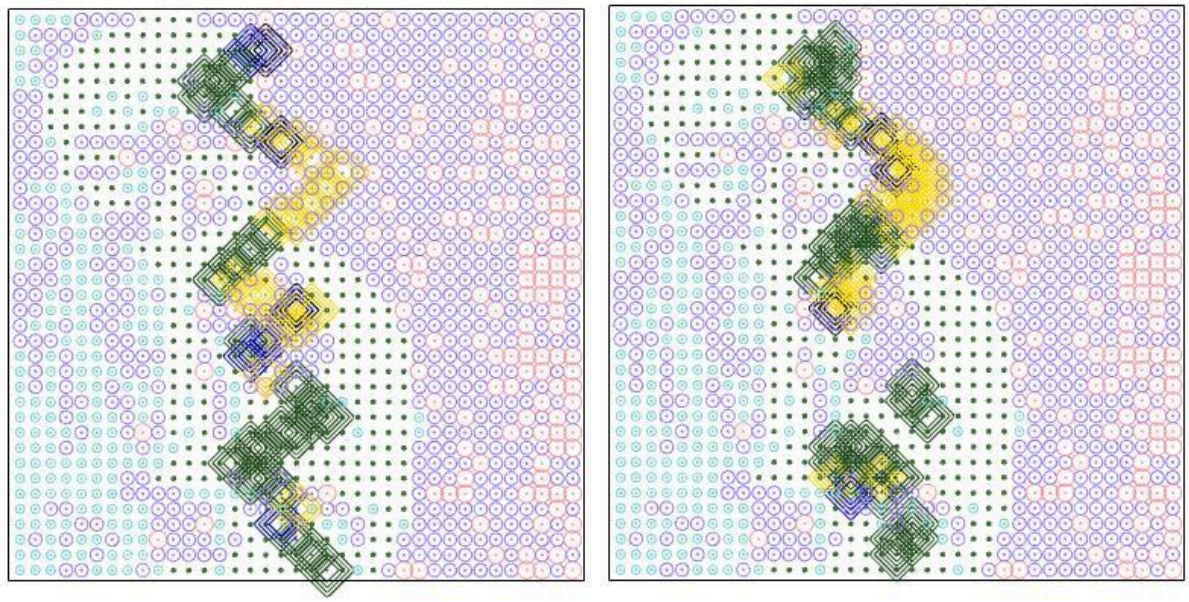


Figure 12. Mode 0: Cellular Automata, with cell size 36.

Figure 13. Mode 1a: Ripple Effect a. (cell size 100) Gif

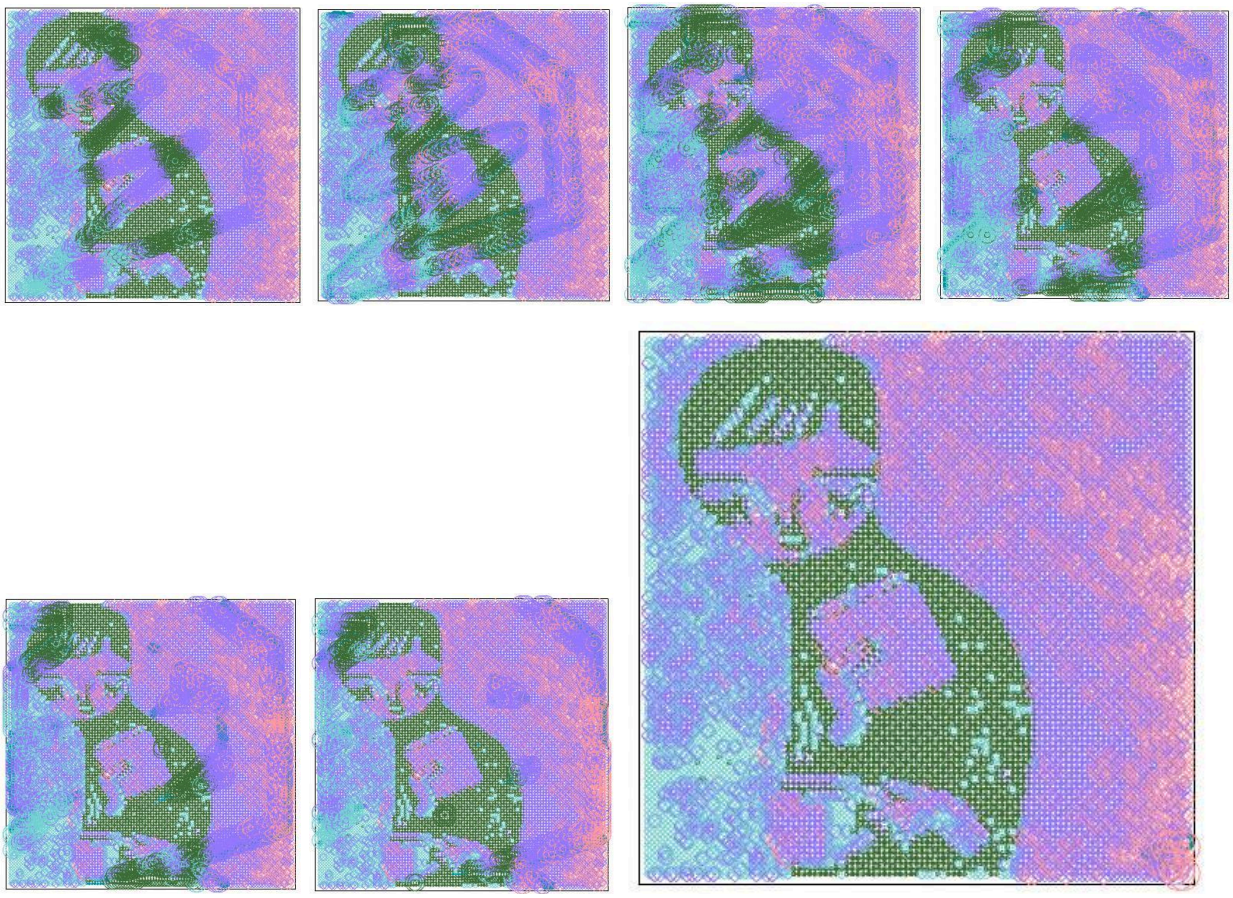


Figure 14. Mode 1b: Ripple Effect b. (cell size 100) Gif

4-2. How does **Mode 1a & 1b: Ripple Effect** work?

This ripple effect, A and B, is imparted after the cellular automata, as I thought it would be nicer to have more than one option. Somehow, the Game of Life doesn't seem to have a crazy effect on me, so I developed two types of ripple effects based on the Python code for the Game of Life. If a neighbor progresses to the live phase, the next phase will either be dead or transformed into an intermediate state. In these modes, I didn't change the color of the active state, as I found the shape conversion created a beautiful blurred effect.

Ripple Effect A presents a rippling visual effect that spreads to the edge, with intermediate and live cells alternating in appearance. The outline of the ripple is turquoise.

Ripple Effect B focuses on the outline of the ripple effect. Once the live cell transitions through an intermediate state, it returns to the dead state.

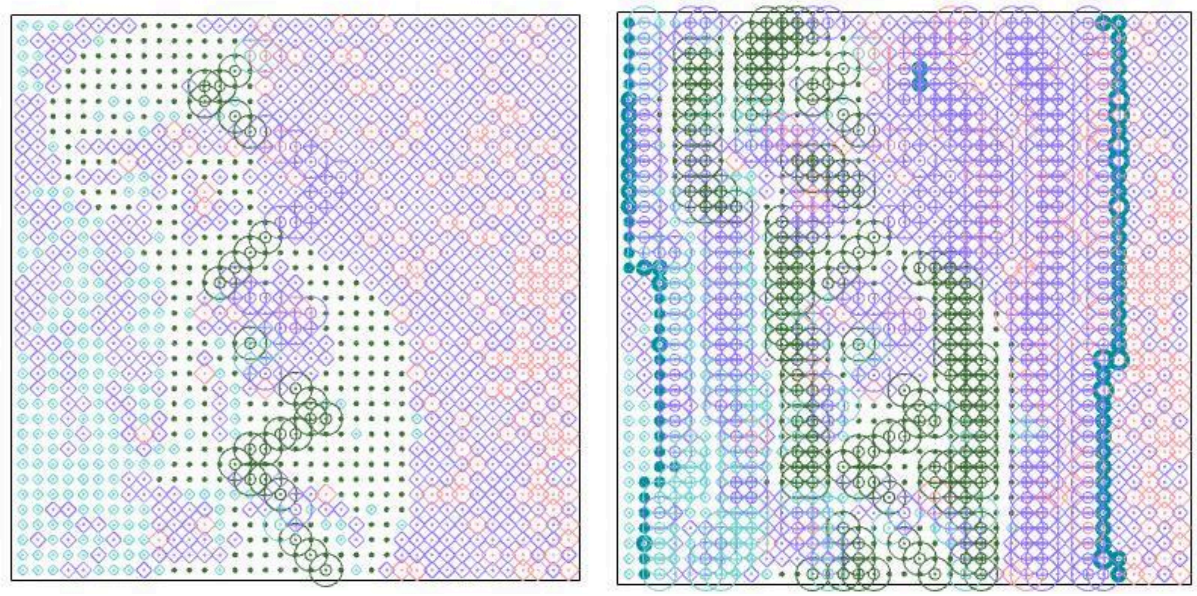


Figure 15. Mode 1a: Ripple Effect a, with cell size 36.

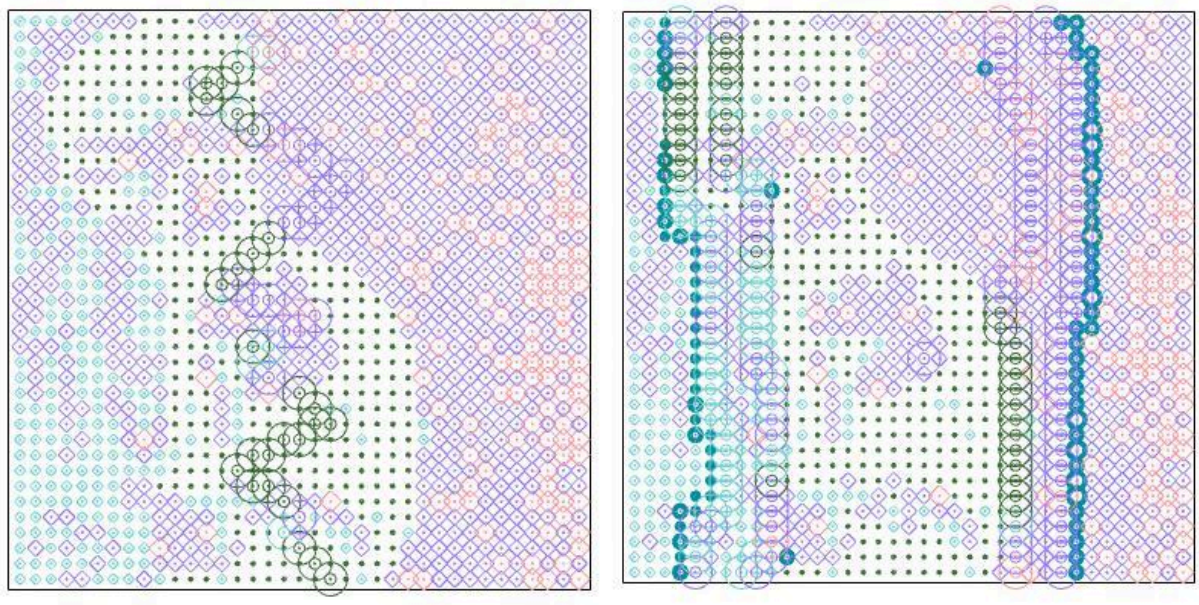


Figure 16. Mode 1b: Ripple Effect b, with cell size 36.

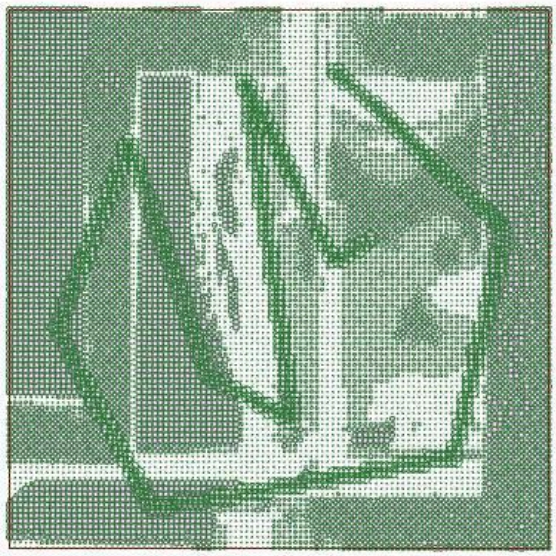
4.3 **Reflection**

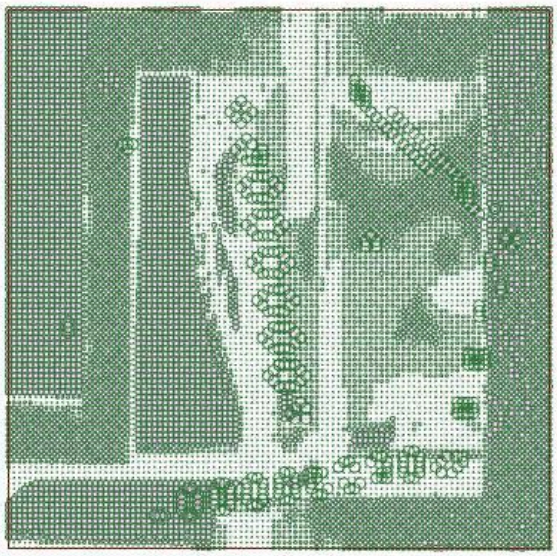
The method for implementing the animation is not a computationally inexpensive approach, as I used Grasshopper remote control panels. It was a bit painful to have the long duration, waiting for the next round. There are limitations with the interface, which helped me figure out the alternative tools in Rhinoceros and Grasshopper. It was a great opportunity to explore the RhinoDeveloper platform. I should drop by more often.

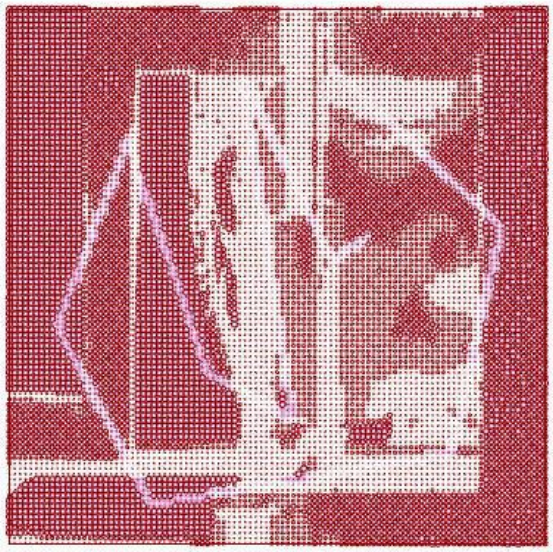
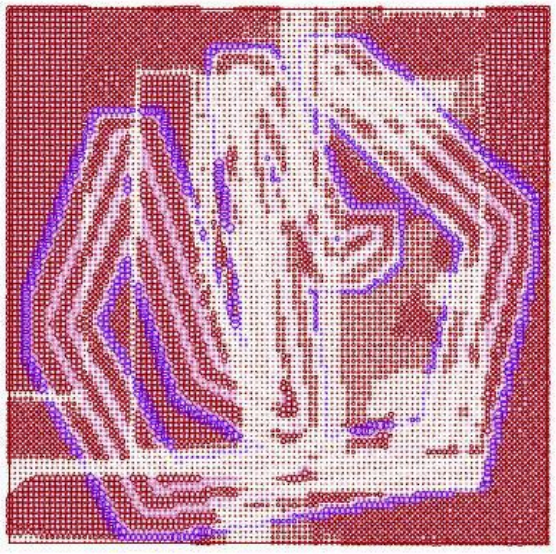
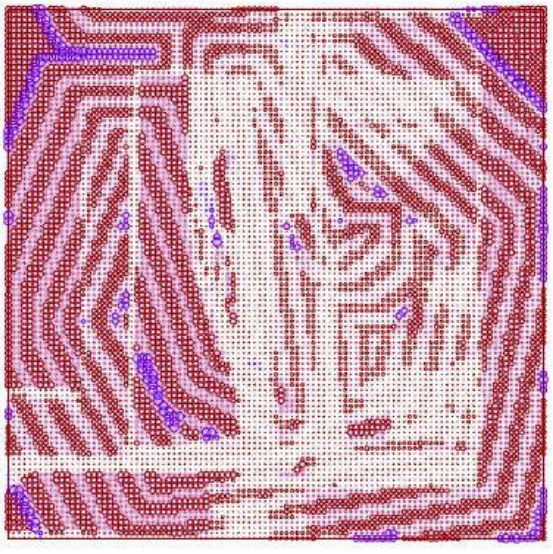
5. References (Bibliography)

48-724 - Scripting and Parametric Design, gameoflifedemo.py, Jimmy Cheng
48-724 - Scripting and Parametric Design, Assignment 6: Cellular Automata, Sohyun Jin
48-724 - Scripting and Parametric Design, UI_Plugin_01.mp4, Jimmy Cheng
Dynamic Drawing of Lines Based on Chosen Points, Steve Baer, McNeel
https://discourse.mcneel.com/t/dynamic-drawing-of-lines-based-on-chosen-points/2199
Simple Grasshopper Animations (Easy), The Different Design
https://www.youtube.com/watch?v=37BsBXaFouY
WISH: Image Sampler with File Path as an INPUT,
https://www.grasshopper3d.com/forum/topics/wish-image-sampler-with-file-path-as-a-input?commentId=2985220%3AComment%3A801927
Grasshopper and MouseEvents in C#, David Rutten
https://www.grasshopper3d.com/forum/topics/grasshopper-and-mouseevents-in?overrideMobileRedirect=1
RhinoCommon API
https://developer.rhino3d.com/api/rhinocommon/

6. Appendix





Please submit your work to Canvas by 11:59 p.m. on December 13. Late submissions will not be accepted.