Creating Variation When Building Image Generation Datasets

By Jhovani Gallardo Moreno, Omar Khan, and **Michael Wehar**

Special thanks to my students and our collaborators E. Brickner, X. Dong, X. Li, C. Liu, J. Mancini, M. Newman-Toker, R. Oet, V. Sumano, L. Suresh, P. Tone, and A. Zhang.

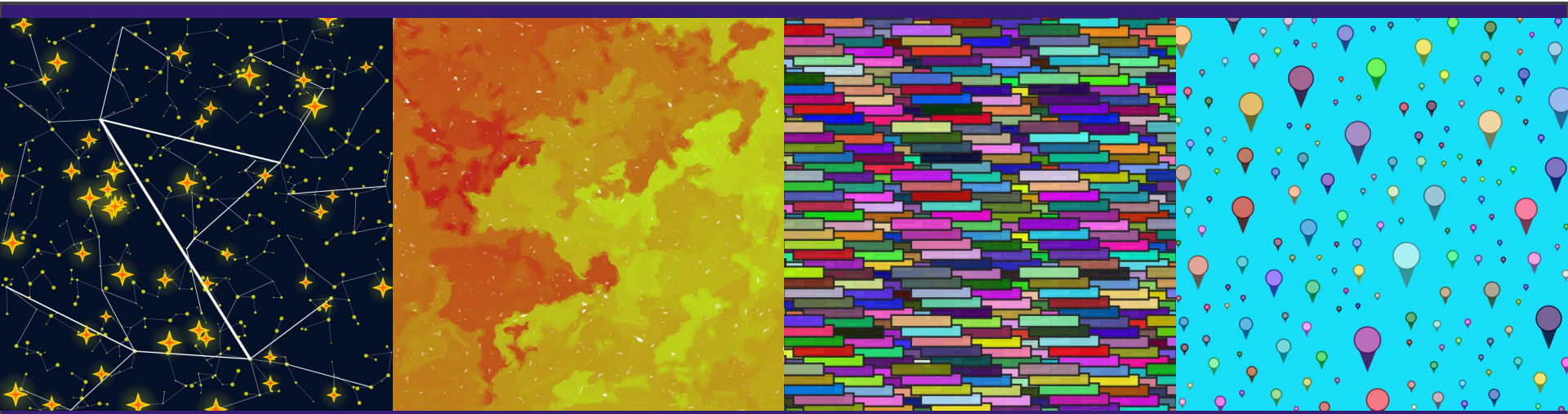


Image Generators

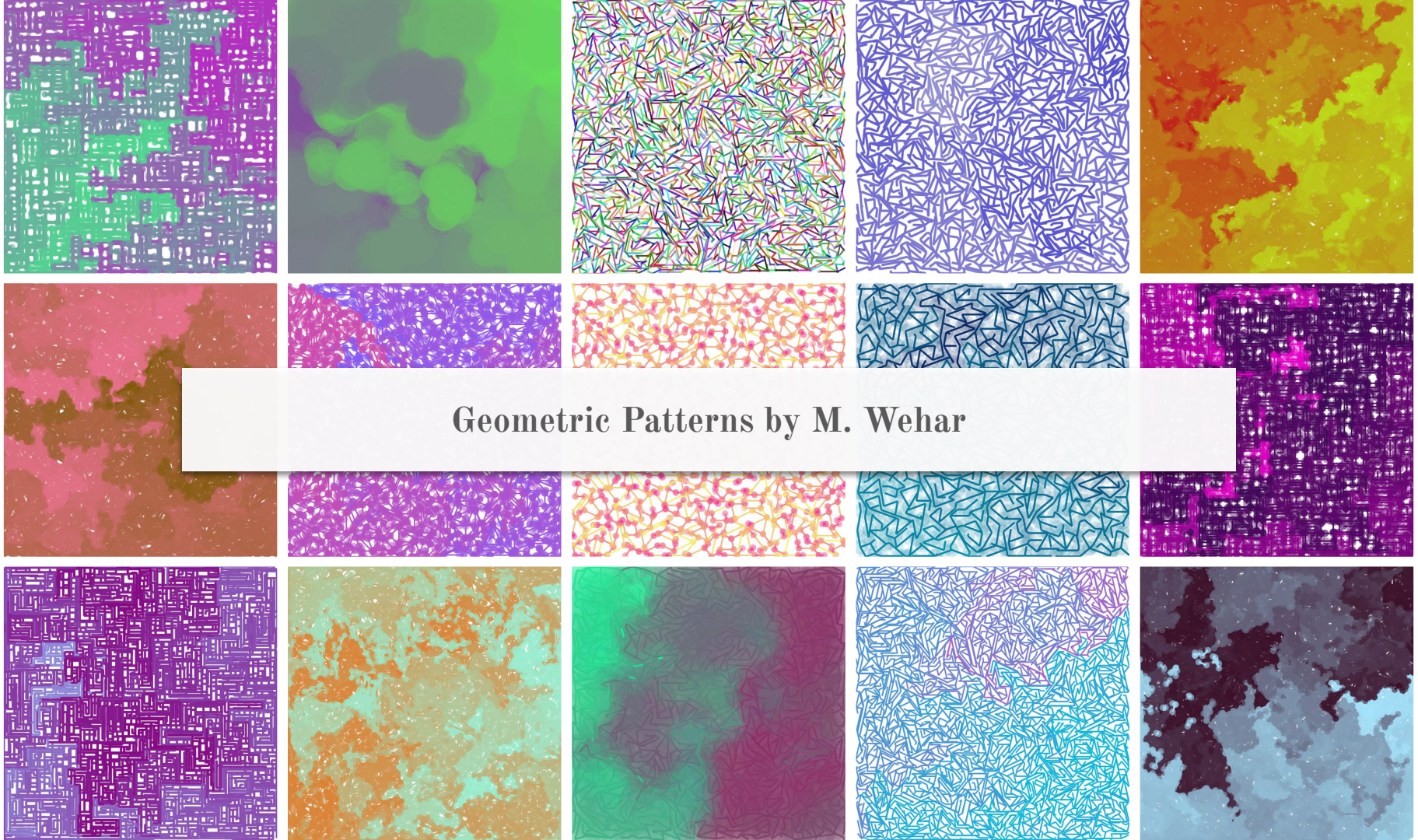
- We are investigating procedural image generation
- Possible examples include:
  - Creating a pattern or artwork based on a predefined algorithm or a set of rules
  - Visualizing a diagram based on a data file
  - Displaying a scene based on predefined models
- Possible use cases: Art, Video Games, Design, Marketing, etc

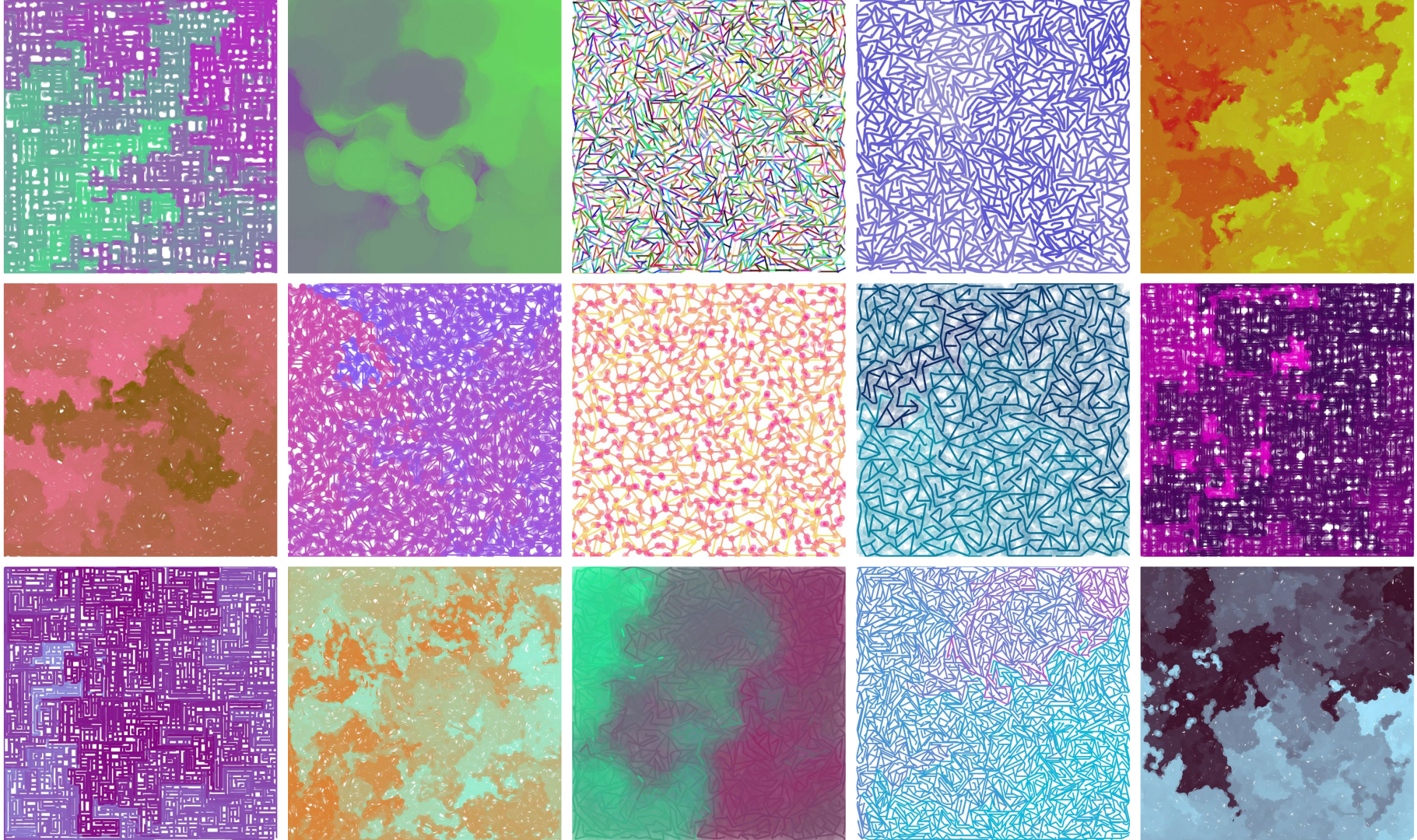
AlgoArt Platform

- **Goal:** Build bridges between computing and art. Offer a low barrier of entry to create procedurally generated art and graphics.
- Web-based Platform consisting:
  - Open Source Creator Studio (on GitHub)
  - Digital Gallery including user reviews and feedback
- Drawing algorithms are written in JavaScript (JS)
  - Every algorithm follows a framework:
    - Methods for **initialize**, **start**, **pause**, **reset**, **drawOneStep**
    - Has a params JS file for customization

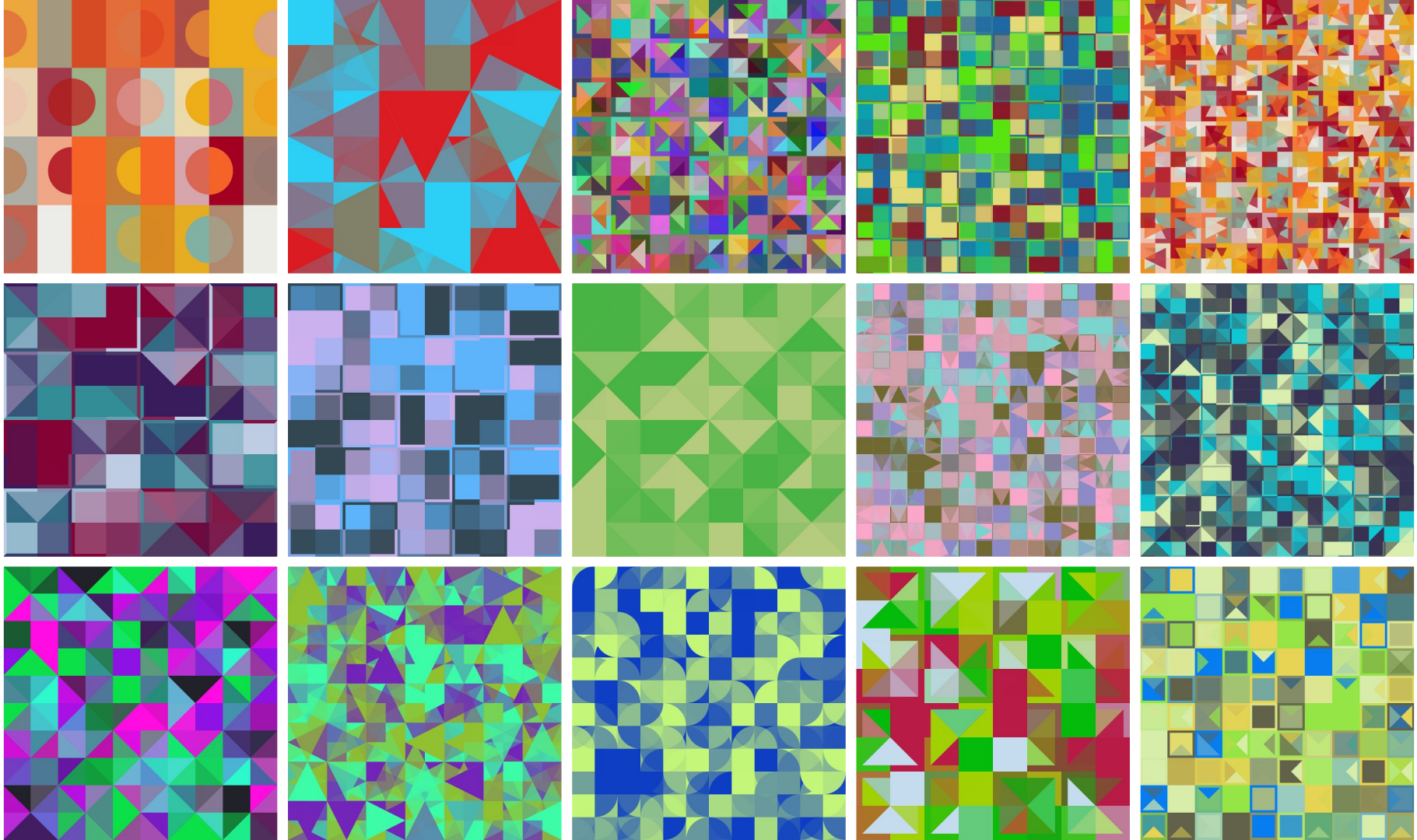
Our Image Data Set

- We generated images using our platform AlgoArt.org
- Developed 15 different drawings algorithms which we used to create 8,000+ unique images
- Received 23,000+ comparison-based reviews
- We hope to expand to more comprehensive reviews in the future to better analyze and understand our dataset


Geometric Patterns by M. Wehar




70’s Funk by A. Zhang and M. Wehar

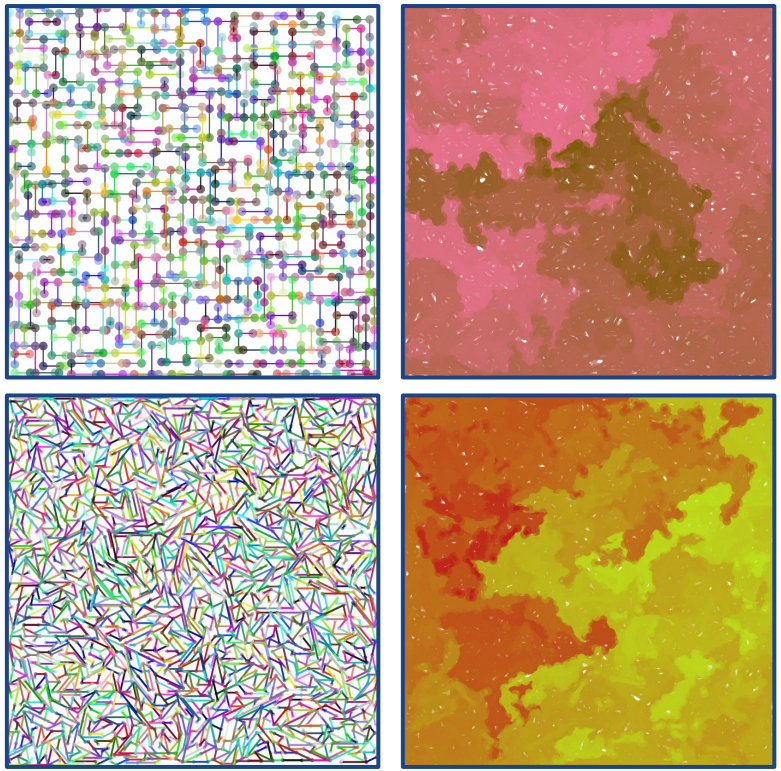


How Do We Incorporate Variation?

- Drawing Algorithms
  - Each algorithm has its own parameters file
  - The parameters manipulate the drawing process
  - Our platform loads the parameters file initially, but the user can modify the params interactively through the UI
- **DEMO** (Creator Studio)

Defining Parameters to Create Variation

- Each algorithm was designed with its own set of parameters
- Although some parameters are very unique to the kind of drawing process, there are generic parameters that apply to most of our drawing algorithms
- Generic Parameter Categories:
  - Colors, Palettes, and Gradients
  - Angles and Uniformity
  - Paths and Movement
  - Sizes and Repetitions



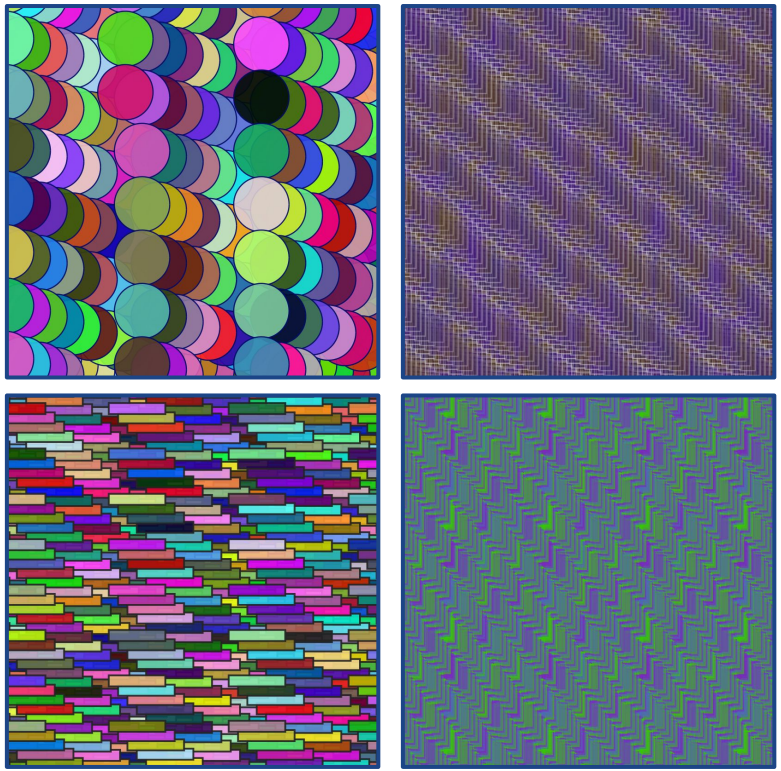
Random Colors (left), Linear Gradient (right)

Developed by M. Wehar

Colors, Palettes, and Gradients (Part 1)

(Geometric Patterns Algorithm)

- Select a starting point
- Randomly sample neighboring points to possibly move to next
- Select one point to move to and connect it together with a line



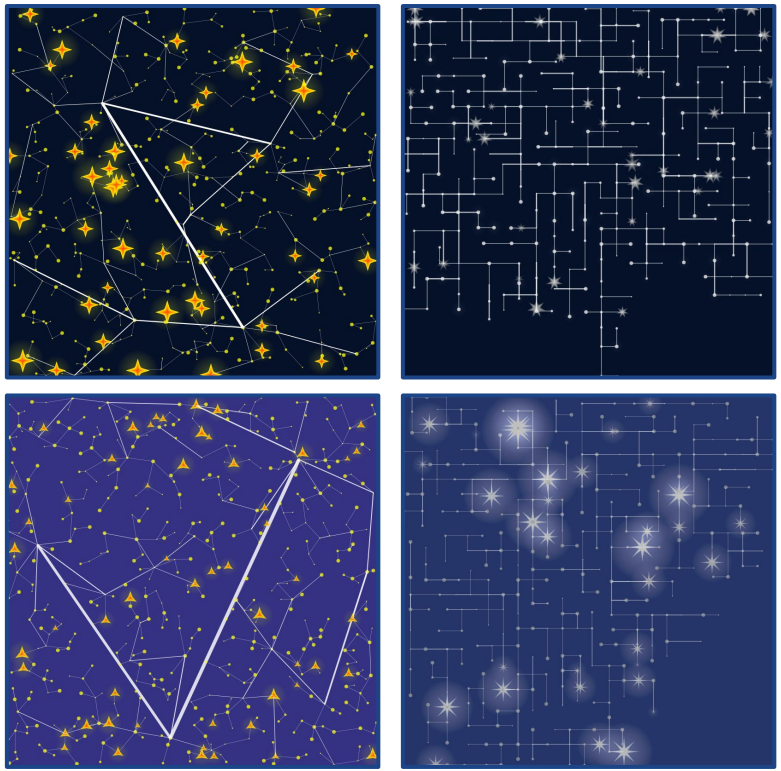
Random Colors (left), Palette (right)

Developed by M. Wehar

Colors, Palettes, and Gradients (Part 2)

(Overlapping Tiles Algorithm)

- Draws a series of grids of tiles on top of each other
- As the tiles layer on top of each other, they blend together creating intricate patterns

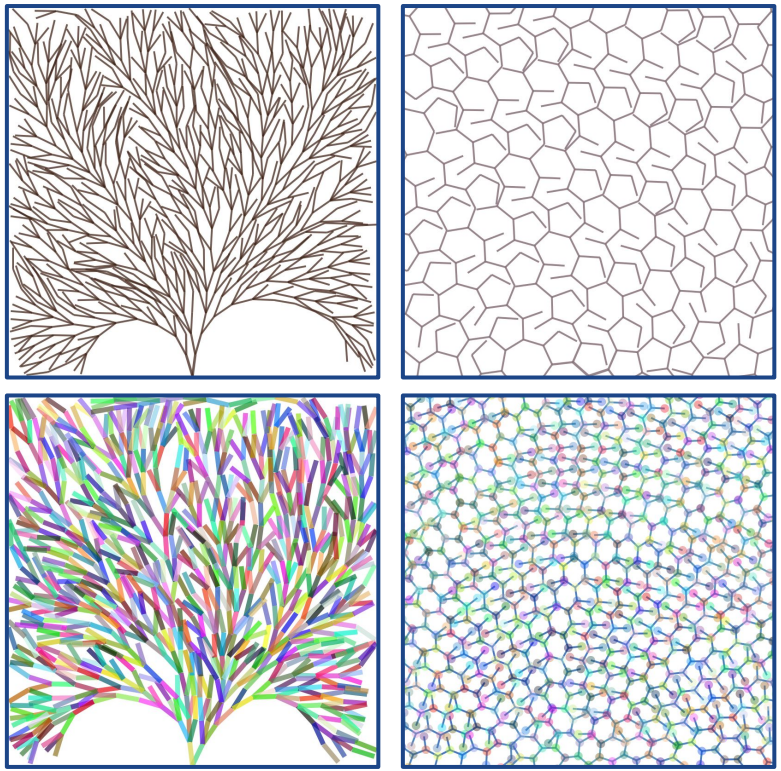


Random Angles (left), Right Angles (right)

Developed by J. Gallardo Moreno

Angles and Uniformity (Part 1)
(Constellations Algorithm)

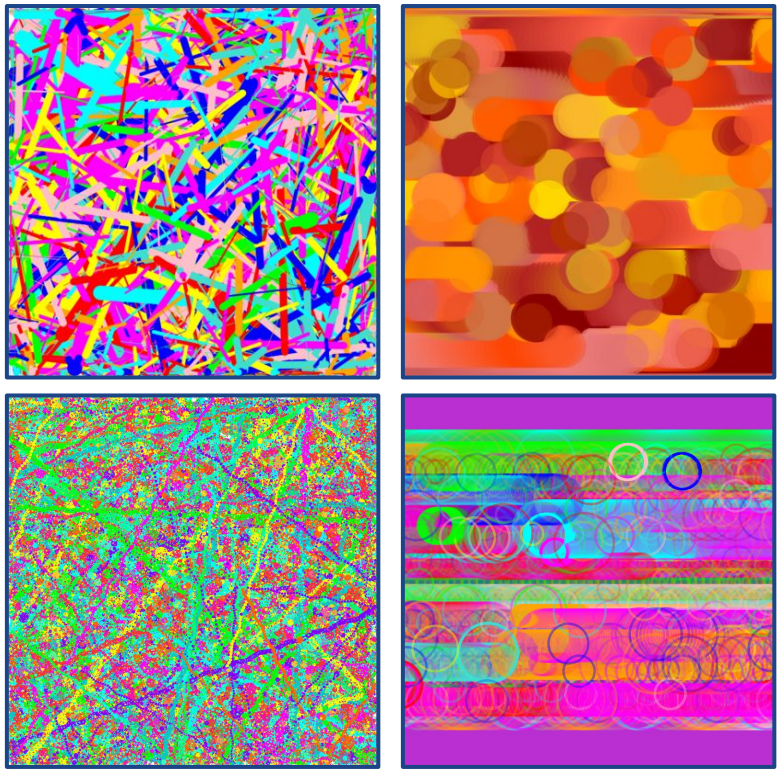
- Builds structures that resemble constellations in the night sky
- Structures are made through a step-by-step process of selecting points and connecting them to the existing structures



Random Angles (left), Repeated Angles (right)
Developed by M. Wehar

Angles and Uniformity (Part 2)
(Trees Algorithm)

- Start with a root node
- Expand outward one step at a time creating many branches
- Trees can resemble real-life or look more artificial depending on parameter choices

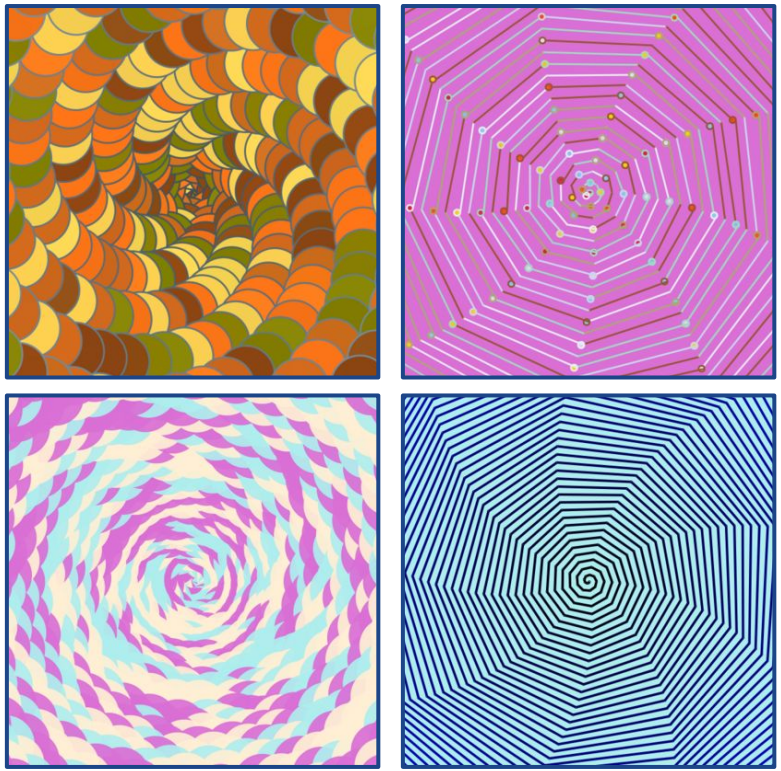


2D Linear Movement (left), Horizontal Only (right)
Developed by O. Khan

Paths and Movement (Part 1)

(Collisions Algorithm)

- Simulates balls moving around and colliding in 2D
- Balls can bounce, combine, or break apart based on parameters
- Conservation of momentum



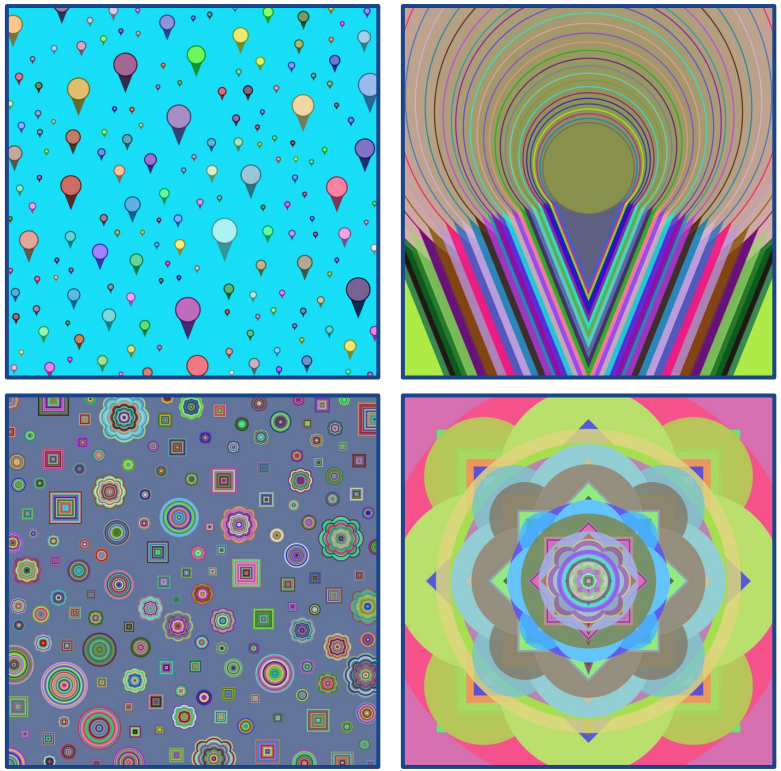
Continuous (left), Discretized (right)

Developed by L. Suresh

Paths and Movement (Part 2)

(Spirals Algorithm)

- Replicates Archimedean Spiral
- Converts polar coordinates to Cartesian coordinates
- Parameters to adjust the spiral style (e.g. discretized)



Multiple Repeated (left), Single Nested (right)
Developed by M. Wehar

Sizes and Repetitions
(Stickers Algorithm)

- Simple sticker shapes and patterns are predesigned
- A combination of sticker patterns are repeatedly placed with different sizes and colors

Dataset Builder and Future Directions

- Our Dataset Builder
  - We have an “engine” called the “dataset builder” where logic can be implemented to randomly sample parameter values to create varying images
- Two Future Directions
  - **Detaching Parameters from Algorithms:** Maybe generic parameters can be automatically added to drawing algorithms? Or, could parameters be used to manipulate the drawing like an image filter, but applied to the drawing operations?
  - **Selecting Parameter Values Based on User Preferences:** Maybe we can sample parameters based on a user’s prior reviews so that we generate images tailored to their preferences?

Thank you!

Visit our GitHub repo

Also, see AlgoArt.org (work in progress!)

We acknowledge support for this work from the Swarthmore College Research and Academic Division Funds.