gTangle

A Grammar for the Procedural Generation of Tangle Patterns

Lab Team

Aman Goel	**CSE**	2018101005
Ammar Ahmed	**CSE**	2018101058
Aryamaan Jain	**LCD**	2019121002
Jyoti Sunkara	**CSE**	2018101044

Mentor **Prajwal Krishna Maitin**

URL **https://github.com/Digital-Image-Processing-IIITH/project-lab-team**

ALL TANGLES FOLLOWING THIS SLIDE HAVE BEEN CREATED

BY THE LAB TEAM

WITH LOVE FOR THE AUDIENCE WATCHING

THIS PRESENTATION HAS BEEN RATED

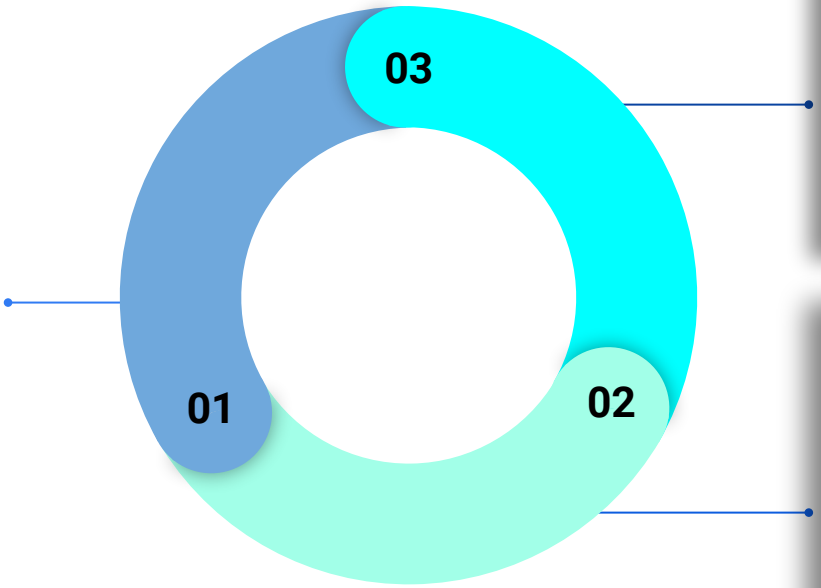
Outputs Drive Link: shorturl.at/gyFI1

Tangles are a form of two dimensional structured pen and ink art created by a small set of basic strokes:

- Dots
- Straight Lines
- Curves
- Shapes

Purpose


THE SIMPLICITY
IT IS BEAUTIFUL

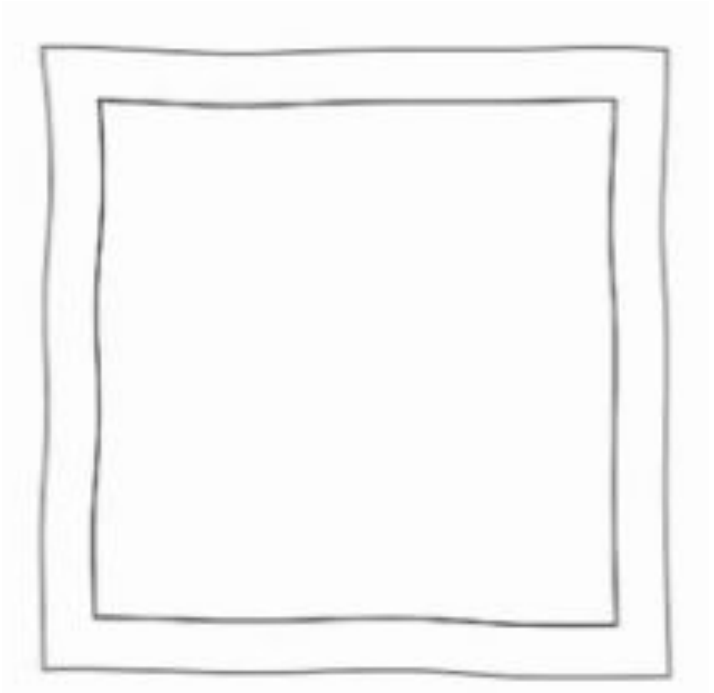

03
01
02


MY PROGRESS IN ART
2008
2017

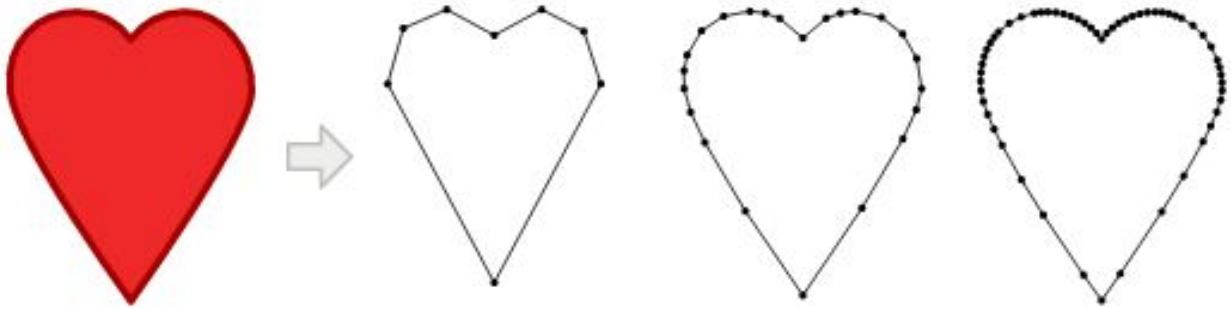

SPEED.
I AM SPEED.

Tangles / Zentangles / Mandalas

The tangles are generated by recursively splitting and combining initial set of polygons, using **group grammars** that perform **operations** on the polygons.



Shapes

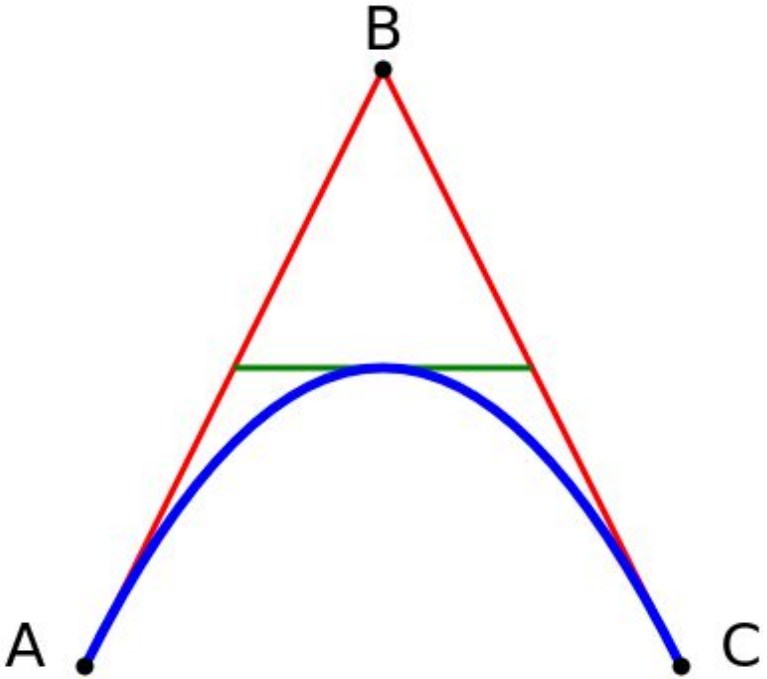


Geometric Shape

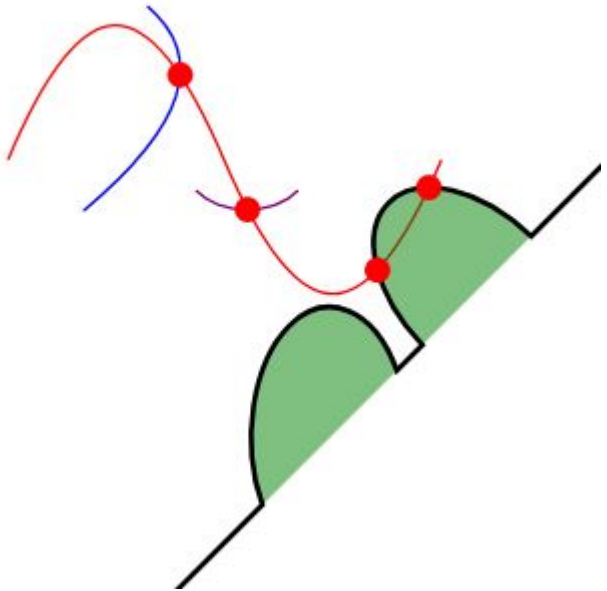
$$S = \langle t, g, s; \Theta, \mathbf{d} \rangle$$

Grammar Shape

Scalable Vector Graphics



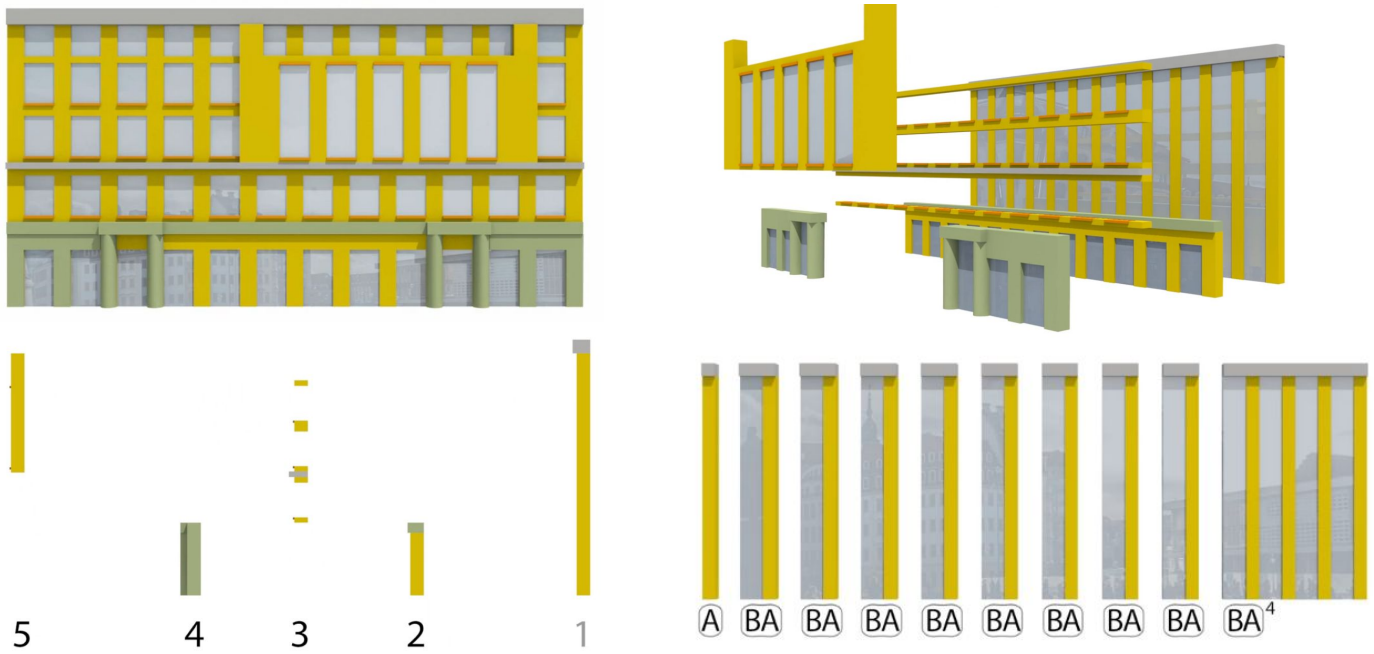
Bézier path



Intersections

Grammar

Layer-Based Procedural Design of Facades, 2015
cg.tuwien.ac.at/research/publications/2015/Ilcik_2015_LAY/



$$R = O(\{p_o\}) : t_m \rightarrow [\langle t_0, g_0 \rangle, \ldots, \langle t_k, g_k \rangle]$$

General Production Rule

JSONs

```
"grammar_name" : "Test Grammar",
"rules" : [
    { "rule_name" : "rule_1",
      "matching_tags" : [],
      "produced_tags" : ["initial_poly"],
      "operator" : "init",
      "parameters" : [0, 600],
      "init_value" : "square"},

    { "rule_name" : "rule_2",
      "matching_tags" : ["initial_poly",],
      "produced_tags" : ["triangles",],
      "operator" : "split",
      "parameters" : [100, 50, 0,], },
```

The grammar here is initializing a square and performing the split operation on the square!

Operators

Grouping Operators

$$ungroup() : t_m \rightarrow [\langle t', g_0 \rangle, \langle t', g_1 \rangle \ldots \langle t', g_k \rangle]$$

$$regroup(k) : t_m \rightarrow [\langle t', g_0 \rangle \ldots \langle t', g_0 \rangle \ldots \langle t', g_k \rangle \ldots \langle t', g_k \rangle]$$

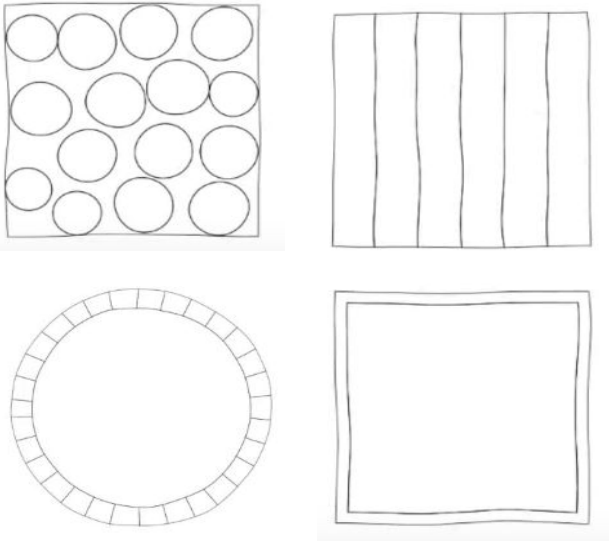
Geometric Operators

$$regularSplit(c, o, x) : t_m \rightarrow [\langle t_0, g_0 \rangle, \ldots, \langle t_0, g_0 \rangle]$$

$$streamlineSplit(type, o) : t_m \rightarrow [\langle t_0, g_0 \rangle, \ldots, \langle t_0, g_0 \rangle]$$

$$outline(d) : t_m \rightarrow [\langle t_{out}, g_0 \rangle, \langle t_{in}, g_1 \rangle, \ldots, \langle t_{in}, g_1 \rangle]$$

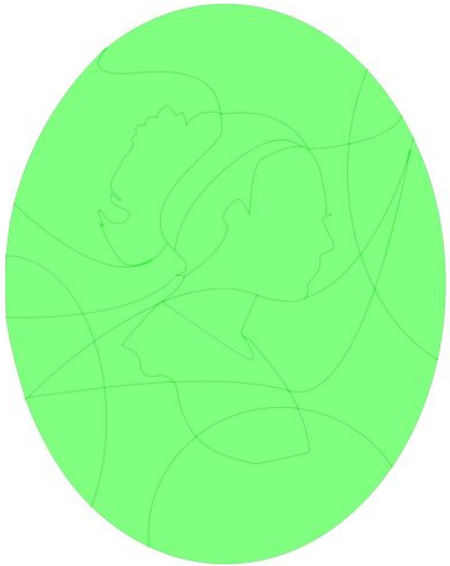
$$place(s, d) : t_m \rightarrow [\langle t_{rem}, g_{rem} \rangle, \langle t_0, g_0 \rangle, \ldots, \langle t_0, g_0 \rangle]$$



Grouping Operators

$$ungroup() : t_m \rightarrow [\langle t', g_0 \rangle, \langle t', g_1 \rangle \ldots \langle t', g_k \rangle]$$

$$regroup(k) : t_m \rightarrow [\langle t', g_0 \rangle \ldots \langle t', g_0 \rangle \ldots \langle t', g_k \rangle \ldots \langle t', g_k \rangle]$$



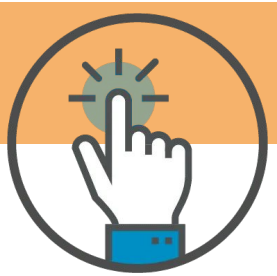
Without Regrouping

Regroup By 1

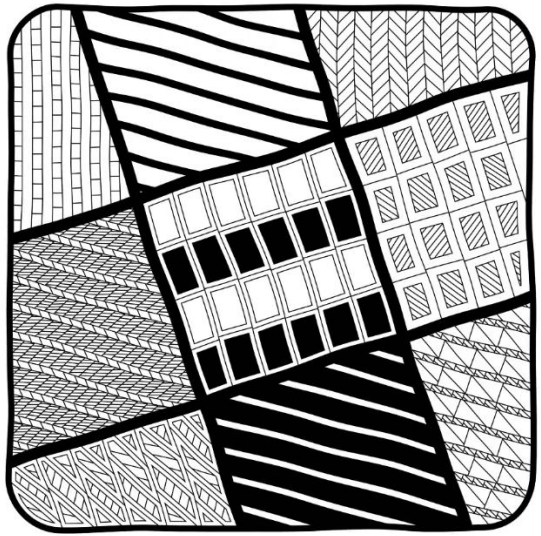
Regroup By 3

Ungroup

Geometric Operators

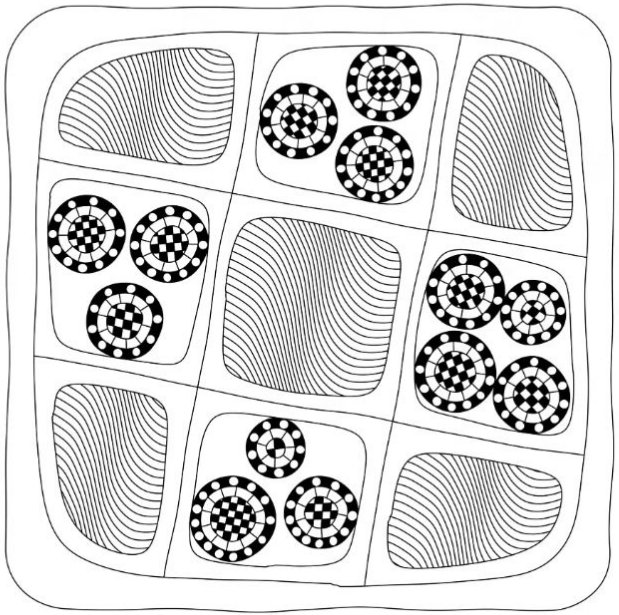


$$regularSplit(c, o, x) : t_m \rightarrow [\langle t_0, g_0 \rangle, \ldots, \langle t_0, g_0 \rangle]$$



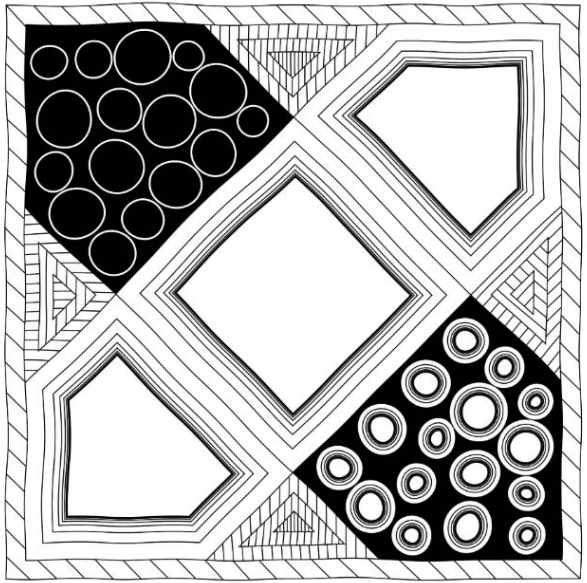
Geometric Operators

$$streamlineSplit(type, o) : t_m \rightarrow [\langle t_0, g_0 \rangle, \ldots, \langle t_0, g_0 \rangle]$$



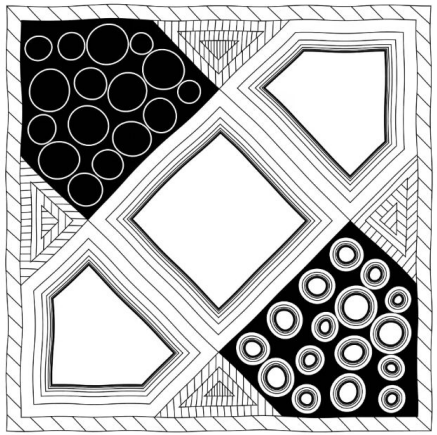
Geometric Operators

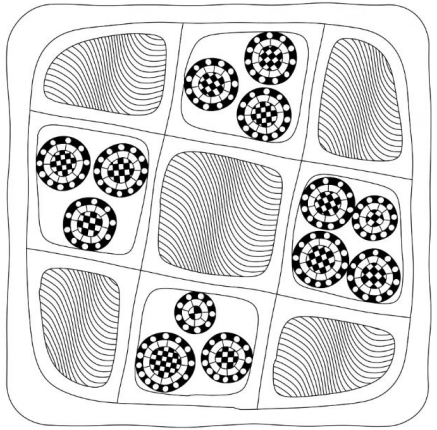
$$outline(d) : t_m \rightarrow [\langle t_{out}, g_0 \rangle, \langle t_{in}, g_1 \rangle, \ldots, \langle t_{in}, g_1 \rangle]$$



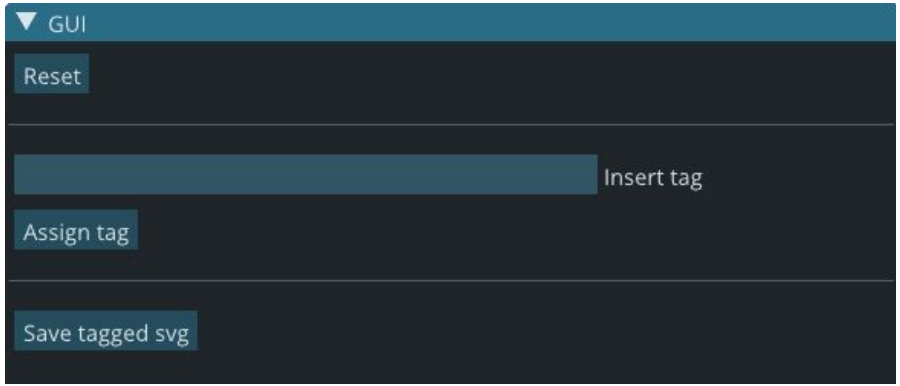
Geometric Operators

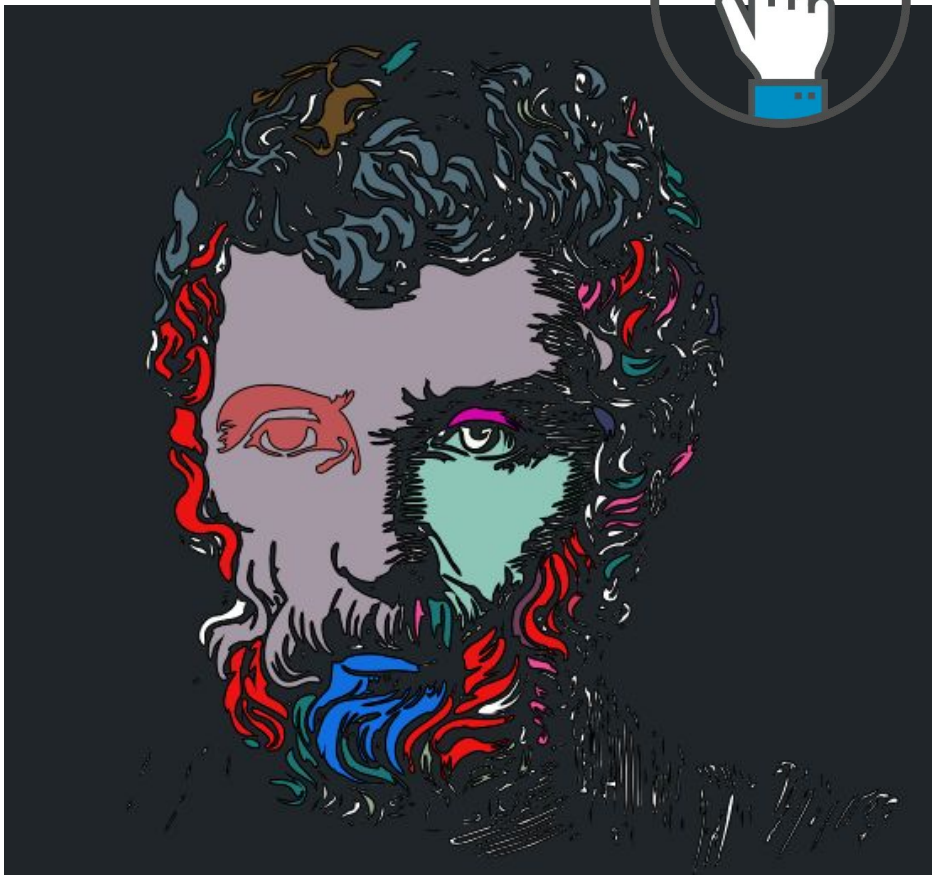
$$place(s, d) : t_m \rightarrow [\langle t_{rem}, g_{rem} \rangle, \langle t_0, g_0 \rangle, \ldots, \langle t_0, g_0 \rangle]$$





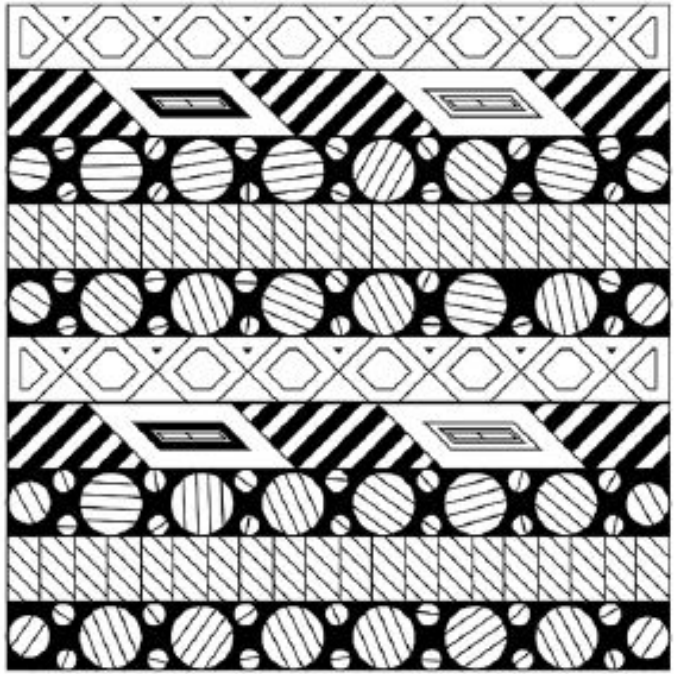
GUI

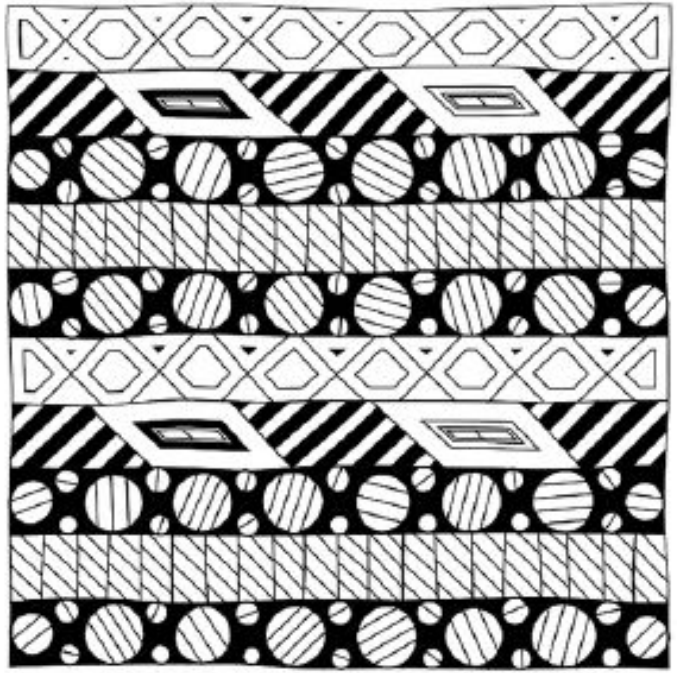


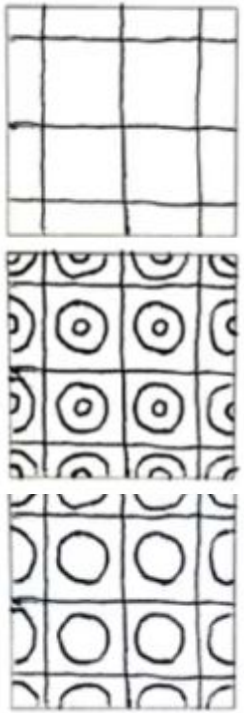


A user interface to select closed areas in the svg and assign a tag to it!

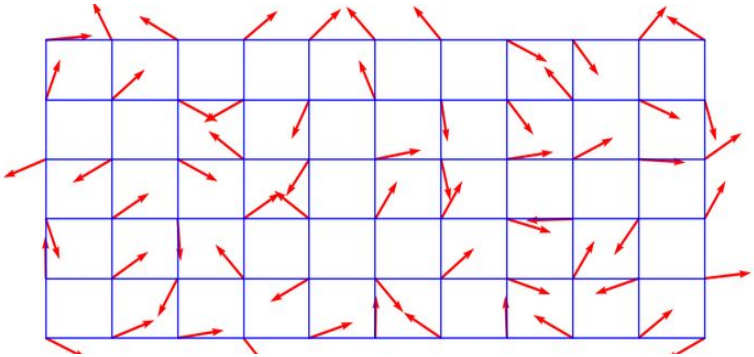
Perturbation

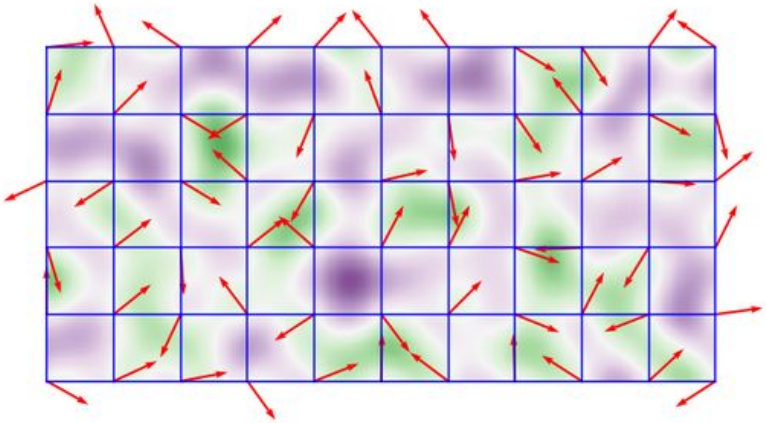


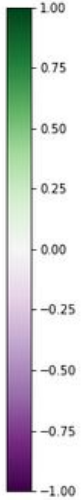


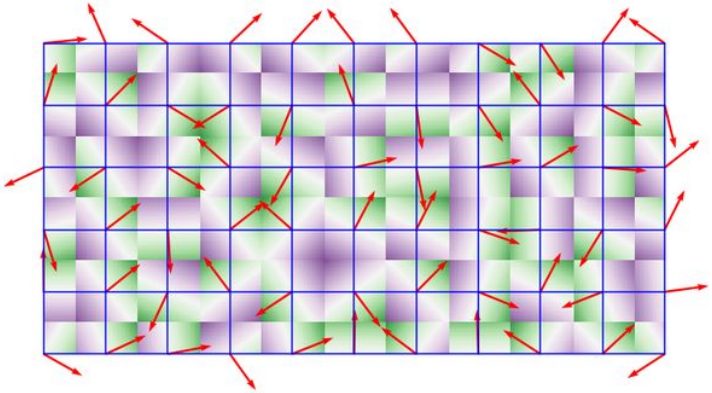


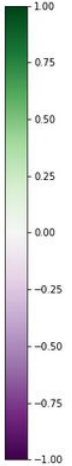
Perlin Noise











$$f(x) = a_0 + \text{smoothstep}(x) \cdot (a_1 - a_0) \text{ for } 0 \leq x \leq 1$$

Iteration

Shape Perturbation

A coherent noise function that is applied to all shapes with the same group id

5

Decorative Operators

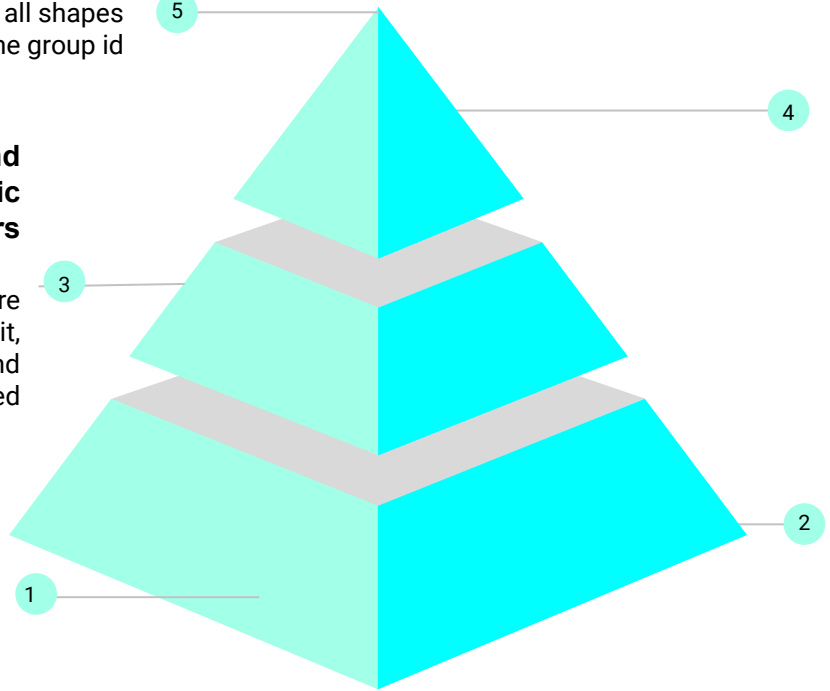
4

The shapes are filled and stippled

Grouping And Geometric Operators

3

The shapes are grouped, split, placed and outlined



Input

An outline sketch in SVG format is provided as input

1

Or an SVG outline is created by us.

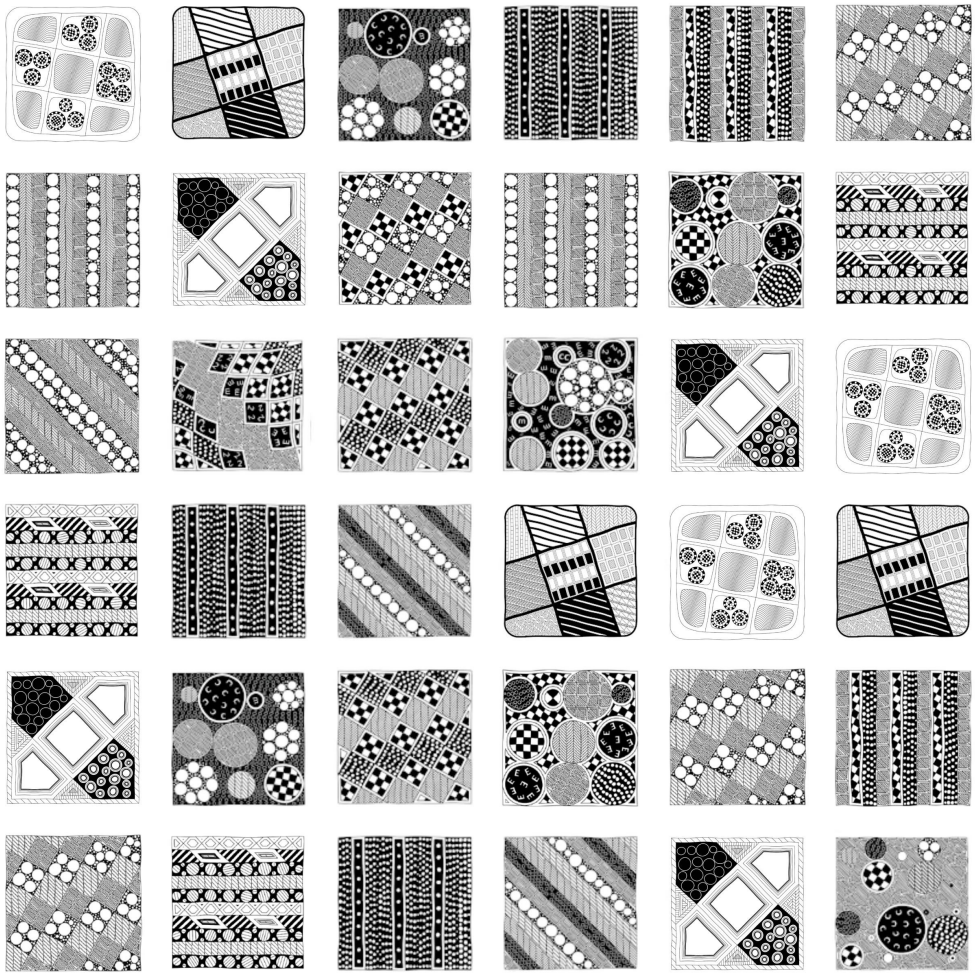
Tree

2

The SVG is stored as a tree data structure after being bisected into the elemental structures

Results

1. $ungroup() : big\_poly \rightarrow [\langle ung\_big\_poly \rangle]$
2. $place(25, 50) : ung\_big\_poly \rightarrow [\langle final \rangle, \langle med\_reminder \rangle]$
3. $place(10, 10) : med\_reminder \rightarrow [\langle final \rangle, \langle final\_reminder \rangle]$
4. $regularSplit(line, 25, 0) : ung\_big\_poly \rightarrow [\langle med\_slice\_poly \rangle]$
5. $regularSplit(line, 10, 0) : med\_slice\_poly \rightarrow [\langle term \rangle]$
6. $outline(10) : ung\_big\_poly \rightarrow [\langle medium3\_poly \rangle, \langle term \rangle]$
7. $regularSplit(line, 25, 1) : medium3\_poly \rightarrow [\langle grid\_poly \rangle]$
8. $invert() : grid\_poly \rightarrow [\langle term \rangle, \langle term \rangle]$
9. $regularSplit(line, 15, 0) : ung\_big\_poly \rightarrow [\langle med\_slice\_poly\_2 \rangle]$
10. $place(13, 20) : med\_slice\_poly\_2 \rightarrow [\langle blobs \rangle, \langle final\_reminder2 \rangle]$
11. $outline(3) : blobs \rightarrow [\langle final\_reminder \rangle, \langle term \rangle]$
12. $regularSplit(line, 20, 0) : ung\_big\_poly \rightarrow [\langle med\_slice\_poly\_3 \rangle]$
13. $regularSplit(line, 20, 0) : med\_slice\_poly\_3 \rightarrow [\langle med\_slice\_poly\_4 \rangle]$
14. $place(13, 20) : med\_slice\_poly\_4 \rightarrow [\langle blobs \rangle, \langle final\_reminder \rangle]$
15. $outline(3) : blobs \rightarrow [\langle final\_reminder \rangle, \langle term \rangle]$
16. $outline(10) : ung\_big\_poly \rightarrow [\langle medium1\_poly \rangle, \langle medium1\_outline \rangle]$
17. $ungroup() : medium1\_poly \rightarrow [\langle ung\_medium1\_poly \rangle]$
18. $place(15, 40) : ung\_medium1\_poly \rightarrow [\langle final \rangle, \langle final\_reminder \rangle]$
19. $regularSplit(line, 45, 1) : ung\_big\_poly \rightarrow [\langle medium2\_poly \rangle]$
20. $regularSplit(line, 30, 0) : medium2\_poly \rightarrow [\langle triangles \rangle]$
21. $regroup(2) : triangles \rightarrow [\langle reg\_triangles \rangle]$
22. $outline(3.6) : reg\_triangles \rightarrow [\langle small\_tri \rangle, \langle outline \rangle]$
23. $regularSplit(line, 5, 0) : small\_tri \rightarrow [\langle term \rangle]$
24. $regularSplit(line, 10, 0) : reg\_triangles \rightarrow [\langle term \rangle]$
25. $invert() : final\_reminder \rightarrow [\langle term \rangle, \langle term \rangle]$



All square tangles on the left have arisen from a single grammar!

Challenges

Challenge 1

Collaboration

With the online semester underway the difficulty of communication, debugging, pair programming and explaining code has become manifold.

Challenge 2

SVG Graphics

SVG defines the graphics of images in XML format.

The intricacies of SVG images, how they are drawn, stored structured and created and how to create them from code were all new problems for us.

Challenge 3

Constructive Solid Geometry Trees

While grammars bring mathematical finesse to the tangles on paper. The representation and manipulation of shapes using a graph in code is a hard problem to solve.

Progress

Extra Work:

- Animation of the entire formation of the tangles allowing the artist to watch the tangle come to life!
- GUI to allow users to tag the SVGs.

Overall:

100%

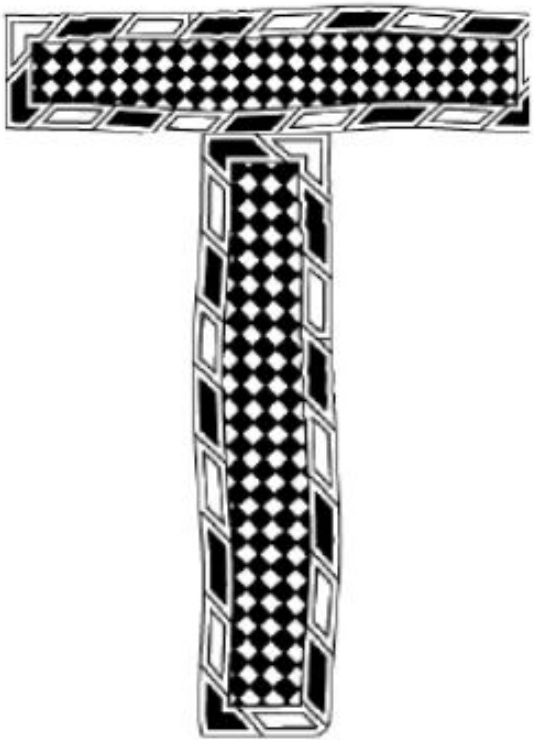
Completed!

The Road Taken

We made it!

Proposal Submission Reading related papers and literature review	Structuring the SVG as a tree. 	Reading and parsing rules as JSON objects	Grouping Operators	Geometric Operators 	Decorative Operators	Shape Perturbation 	Testing And Creating Tangles!

Of these two letters which one was made by us?



Thank you!

Division Of Work

Aman Goel	Coding of operators
Ammar Ahmed	Coding of operators, testing
Aryamaan Jain	Coding of csg tree, documentation
Jyoti Sunkara	Coding operators, csg tree, noise, parsing grammar, generated outputs, documentation