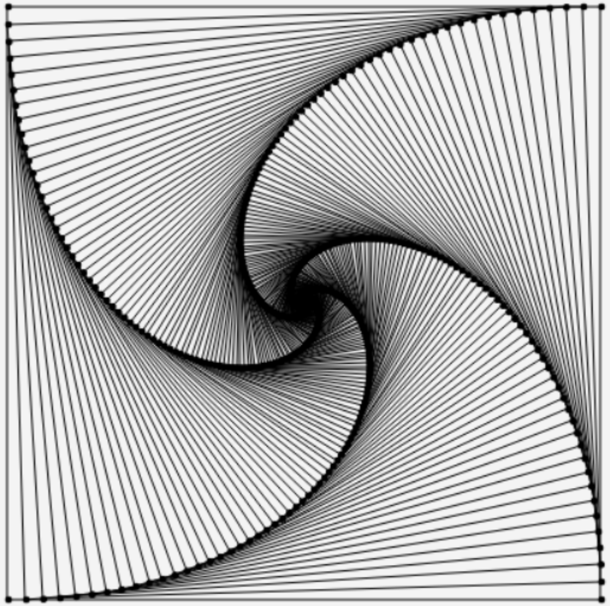
Writing a Shape Grammar Interpreter

Bachelor Thesis

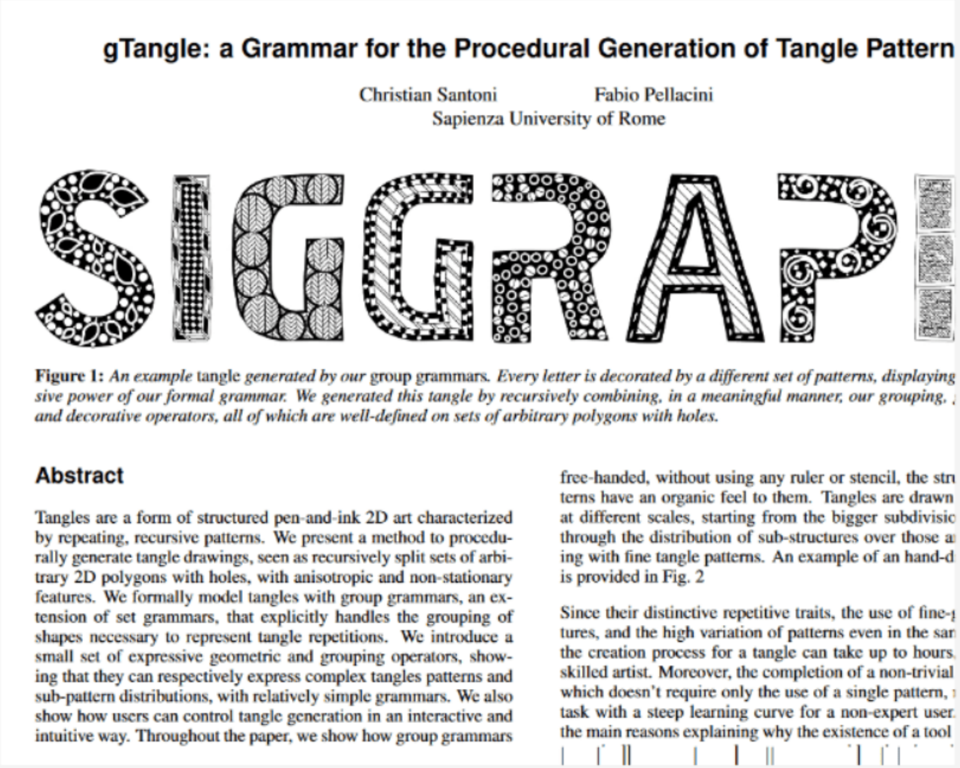
Lars Wüthrich

February 2018



Thesis Idea

- Interesting paper that references shape grammars



gTangle: a Grammar for the Procedural Generation of Tangle Pattern

Christian Santoni Fabio Pellacini
Sapienza University of Rome

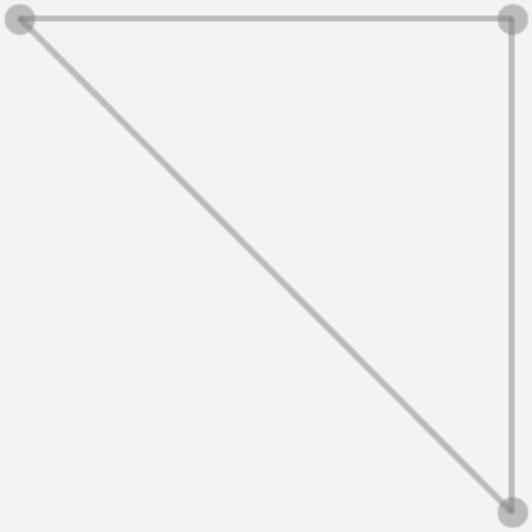
Figure 1: *An example* tangle *generated by our* group grammars*. Every letter is decorated by a different set of patterns, displaying sive power of our formal grammar. We generated this tangle by recursively combining, in a meaningful manner, our grouping, and decorative operators, all of which are well-defined on sets of arbitrary polygons with holes.*

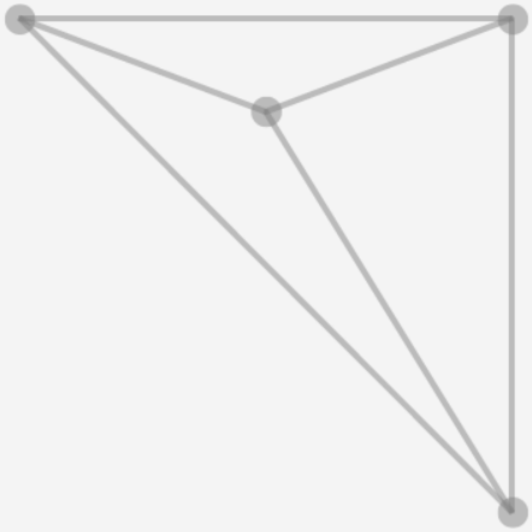
Abstract

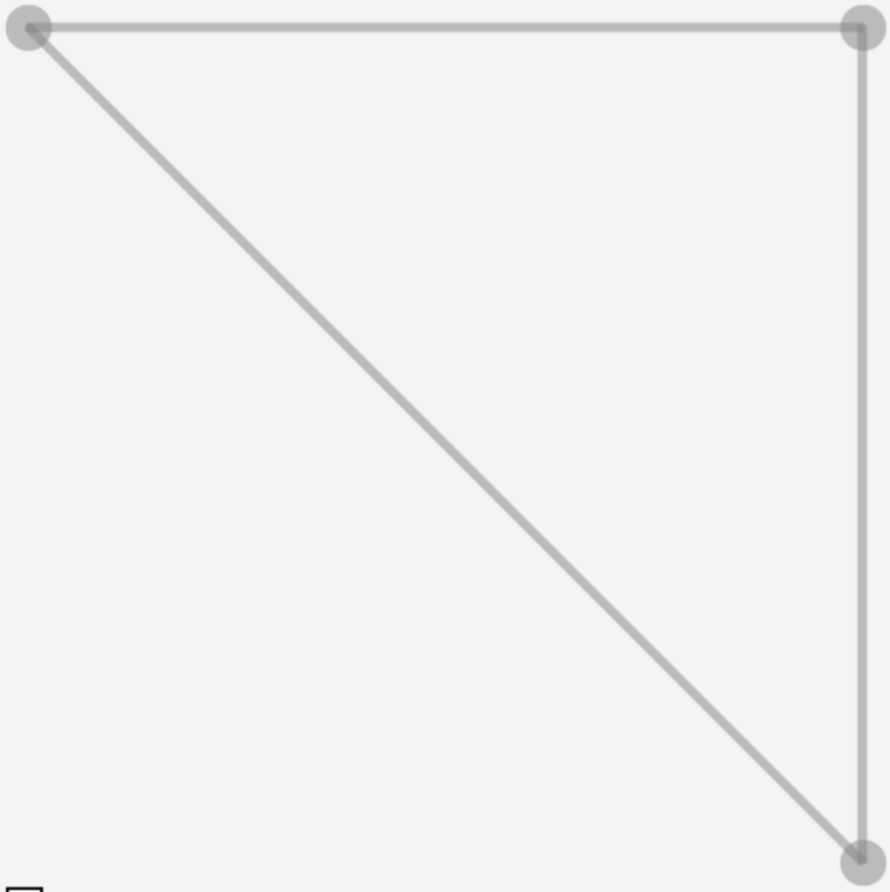
Tangles are a form of structured pen-and-ink 2D art characterized by repeating, recursive patterns. We present a method to procedurally generate tangle drawings, seen as recursively split sets of arbitrary 2D polygons with holes, with anisotropic and non-stationary features. We formally model tangles with group grammars, an extension of set grammars, that explicitly handles the grouping of shapes necessary to represent tangle repetitions. We introduce a small set of expressive geometric and grouping operators, showing that they can respectively express complex tangles patterns and sub-pattern distributions, with relatively simple grammars. We also show how users can control tangle generation in an interactive and intuitive way. Throughout the paper, we show how group grammars

free-handed, without using any ruler or stencil, the str terns have an organic feel to them. Tangles are drawn at different scales, starting from the bigger subdivisio through the distribution of sub-structures over those a ing with fine tangle patterns. An example of an hand-d is provided in Fig. 2

Since their distinctive repetitive traits, the use of fine- tures, and the high variation of patterns even in the sa the creation process for a tangle can take up to hours skilled artist. Moreover, the completion of a non-trivial which doesn't require only the use of a single pattern, task with a steep learning curve for a non-expert user the main reasons explaining why the existence of a tool







intersection

rotation

reflection

present points

reset

map

add new rule

reset

remove rule

reset shape

next match

apply rule

2

The Interpreter

- Subshape Detection - Find all subshapes
- Subshape Selection - Choose one among all subshapes
- Shape Transformation - Apply the rule

Subshape Detection

- Find a transformation τ
- Applying τ on a shape makes it a subshape
- Existing algorithm: The construction of shapes - Krishnamurti 1981
- My algorithm is based on local coordinate point comparison

Homogeneous Coordinates

- Form a projective space
- 3D points have 4 components
- 2D points have 3 components
- We can differentiate between points in 2D:

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

- and vectors (or points at infinite distance):

$$\begin{pmatrix} x \\ y \\ 0 \end{pmatrix}$$

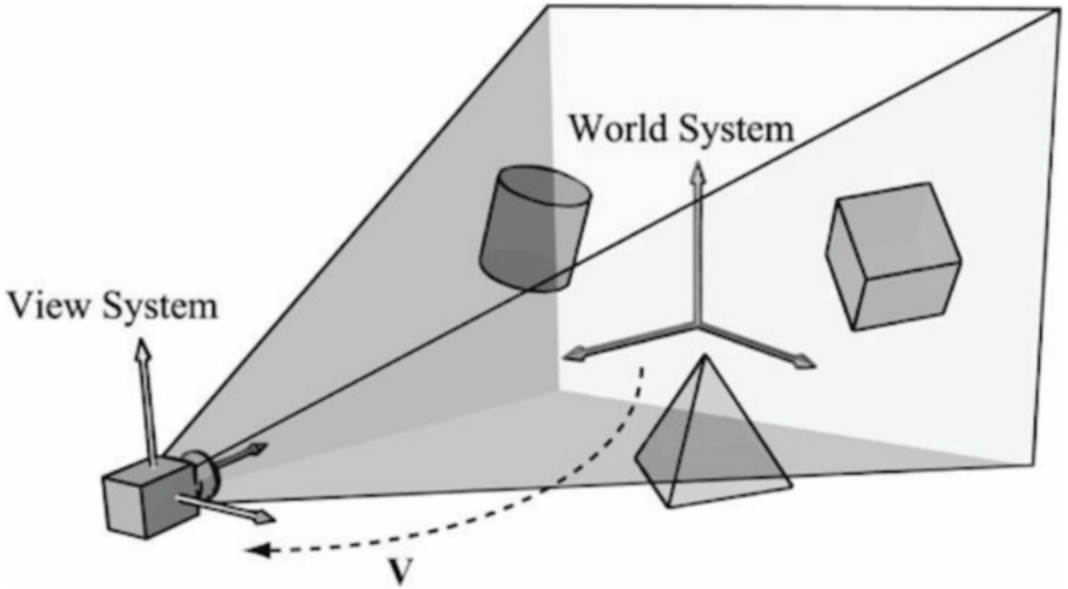


Figure 5.19. **Convert the coordinates of vertices relative to the world space to make them relative to the camera space.**

http://ycpcs.github.io/cs470-fall2014/labs/lab07-2.html

Rotation, Scaling, and Translation in 2D

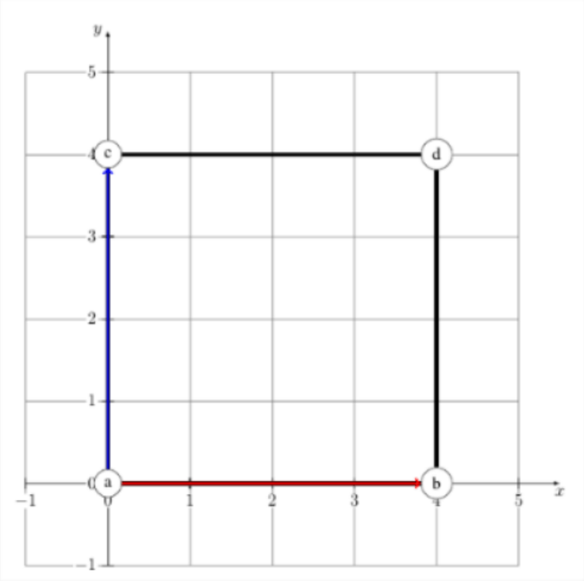
$$\begin{pmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} cos(\theta) \cdot x - sin(\theta) \cdot y \\ sin(\theta) \cdot x + cos(\theta) \cdot y \\ 1 \end{pmatrix}$$

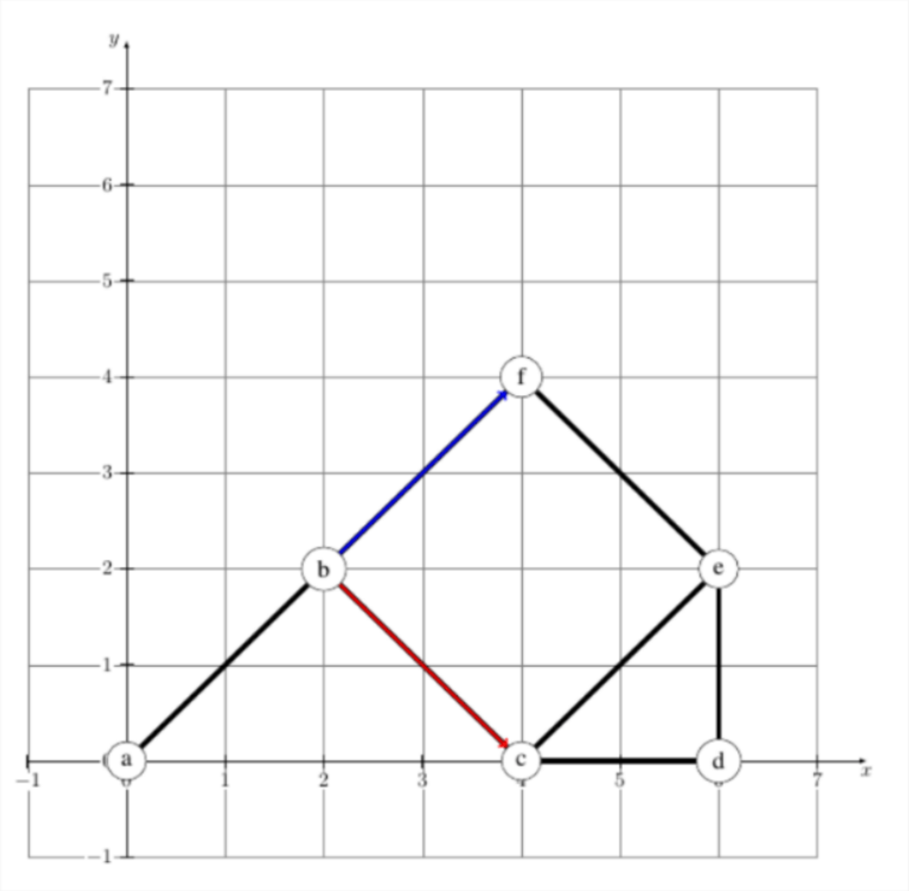
$$\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} a \cdot x \\ b \cdot y \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & a \\ 0 & 1 & b \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x + a \\ y + b \\ 1 \end{pmatrix}$$

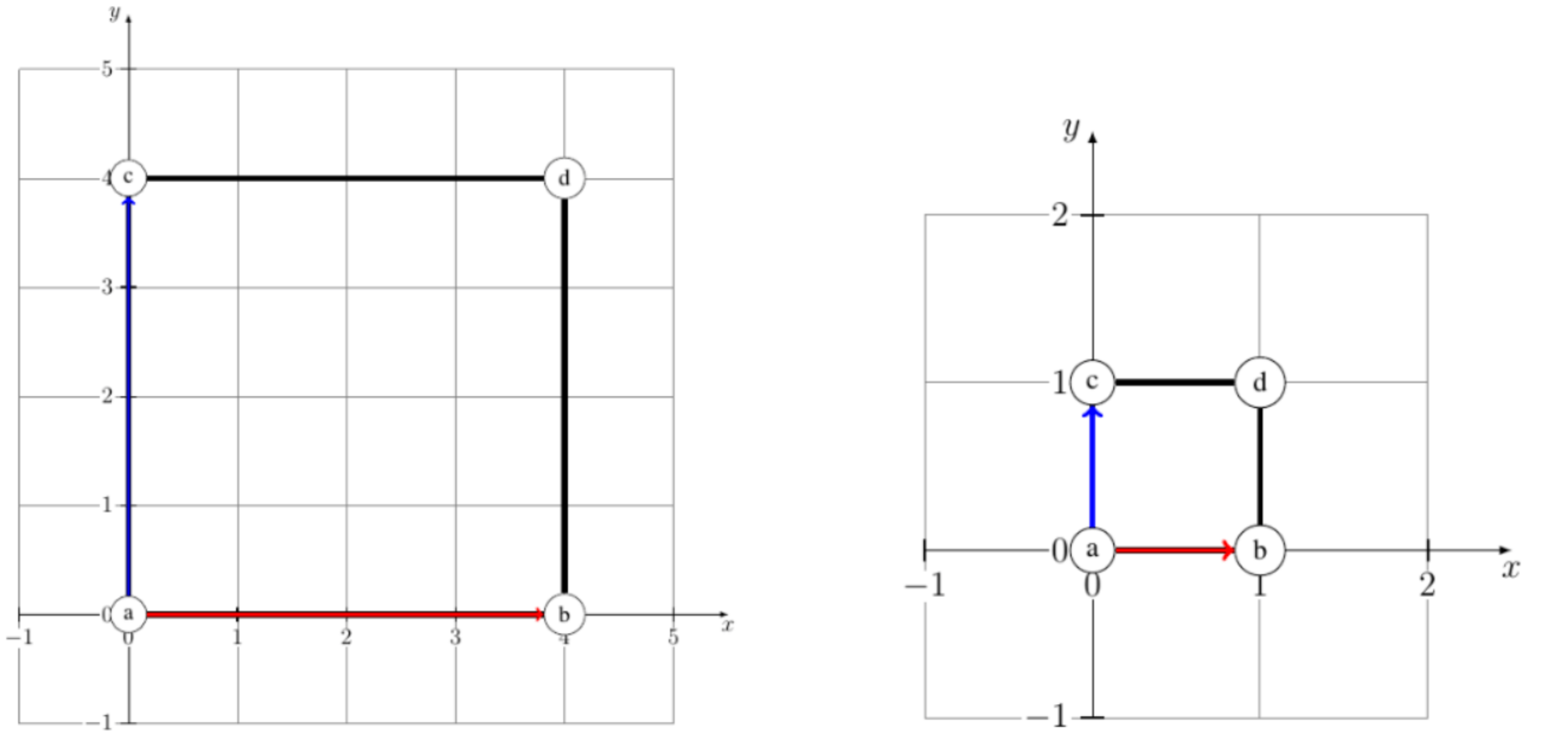
$$\begin{pmatrix} 1 & 0 & a \\ 0 & 1 & b \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 0 \end{pmatrix} = \begin{pmatrix} x \\ y \\ 0 \end{pmatrix}$$

Subshape Detection Example





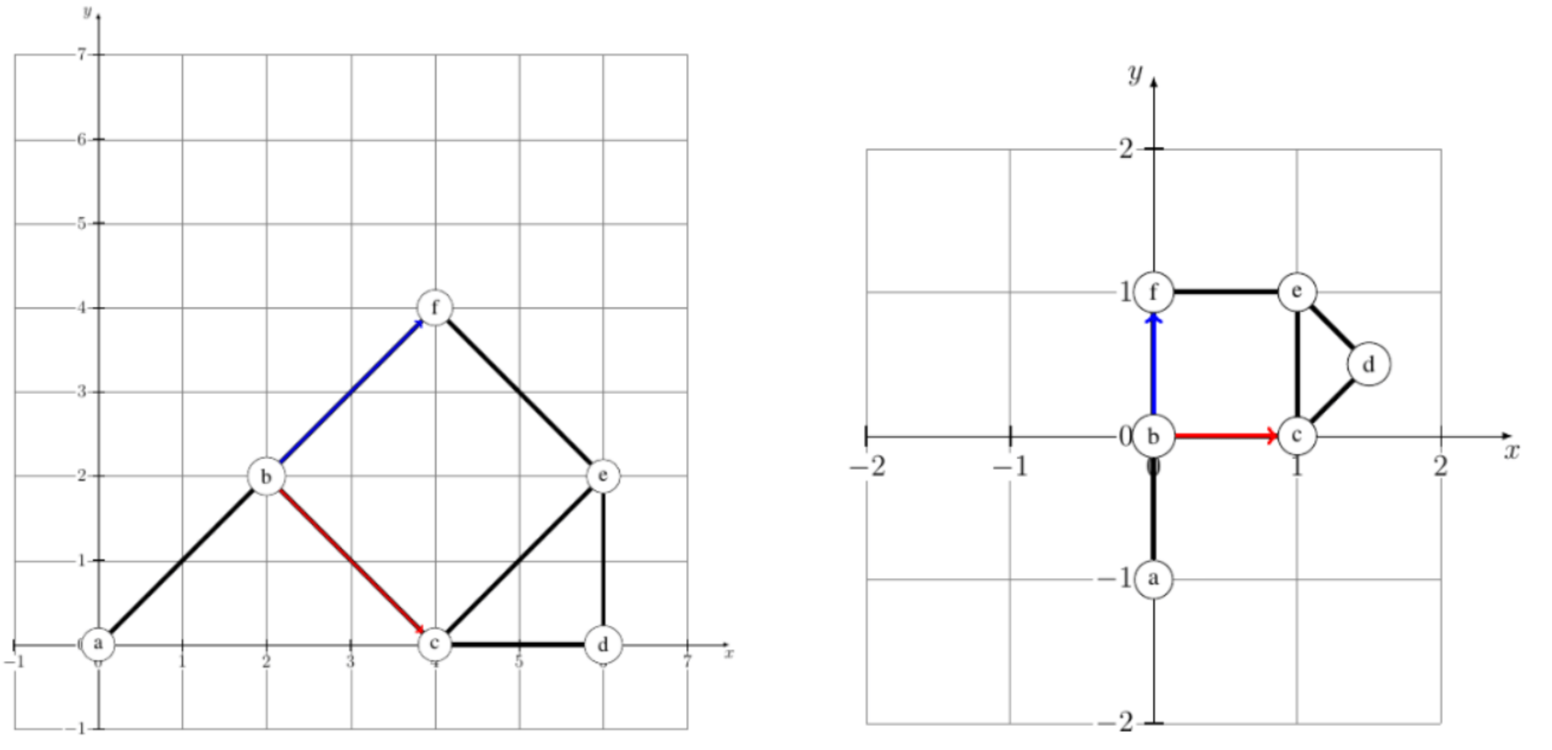
Create a coordinate system in potential subshape



(a) α shape in original coordinate system, yet untransformed

(b) α shape in local coordinates (scaled by $1/4$ in x and y direction)

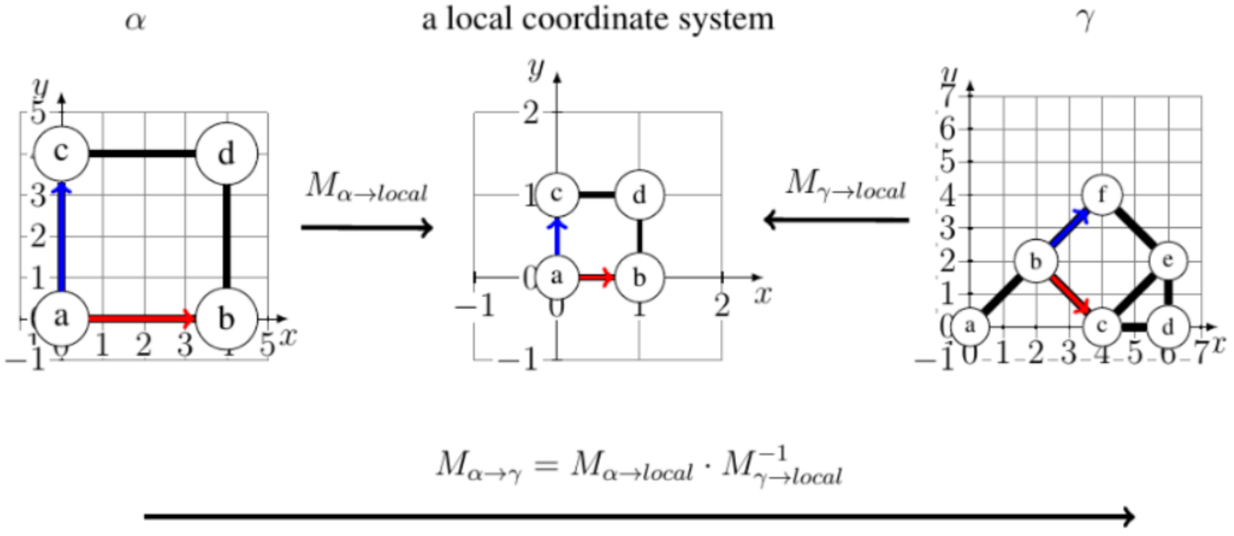
Create a coordinate system in target shape



(a) γ in original coordinate system

(b) γ in new local coordinate system

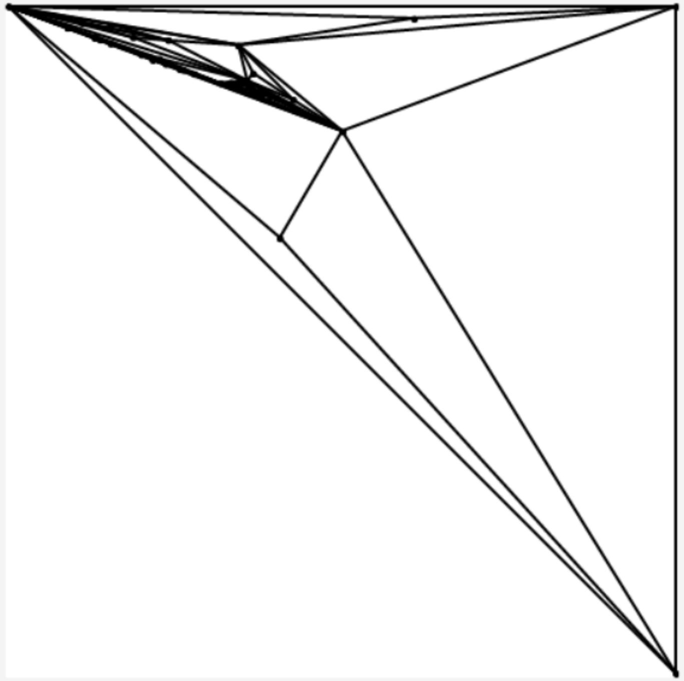
Point comparison in local coordinates



Subshape Selection Problem

- Which triangle should we choose?
- First try - Choose randomly

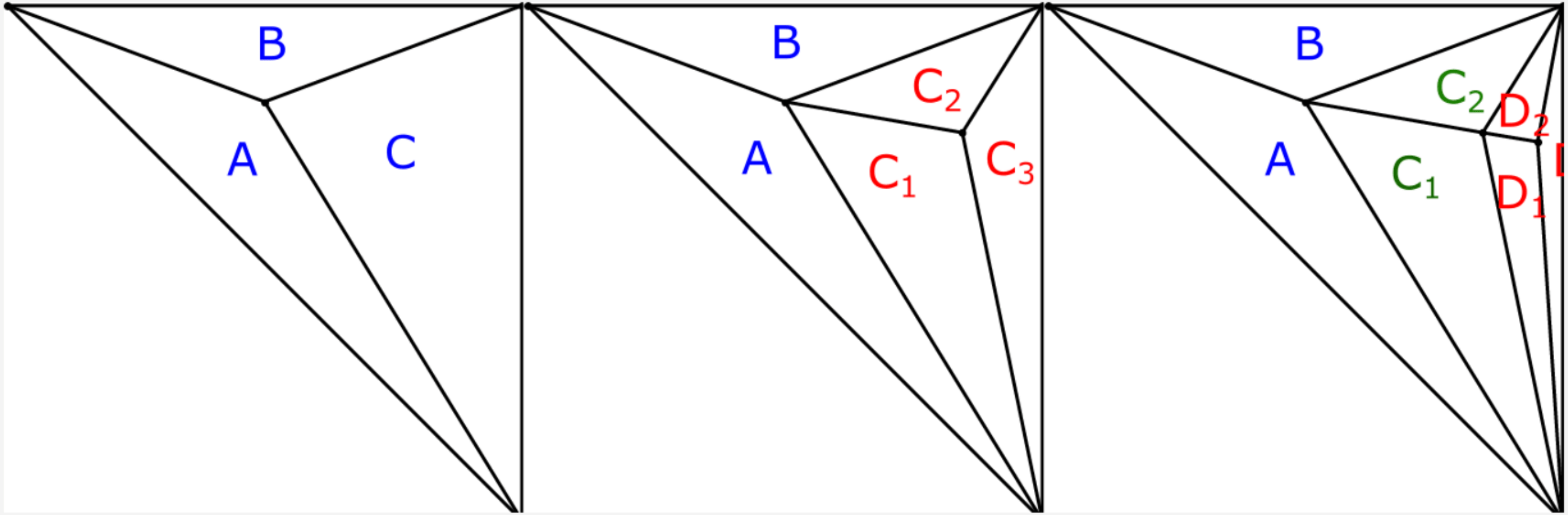
Random Choice Result



13

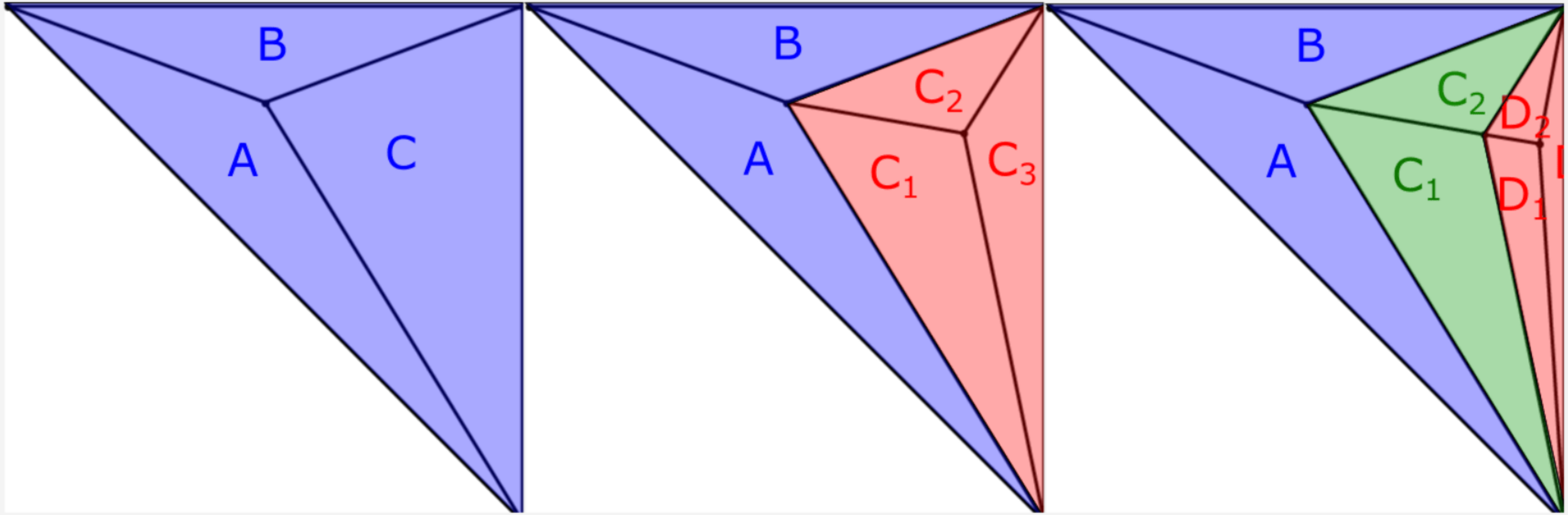
Problem with Random Choice

- Probability of choosing an "older" triangle decreases
- Probability to expand in already split triangles increases

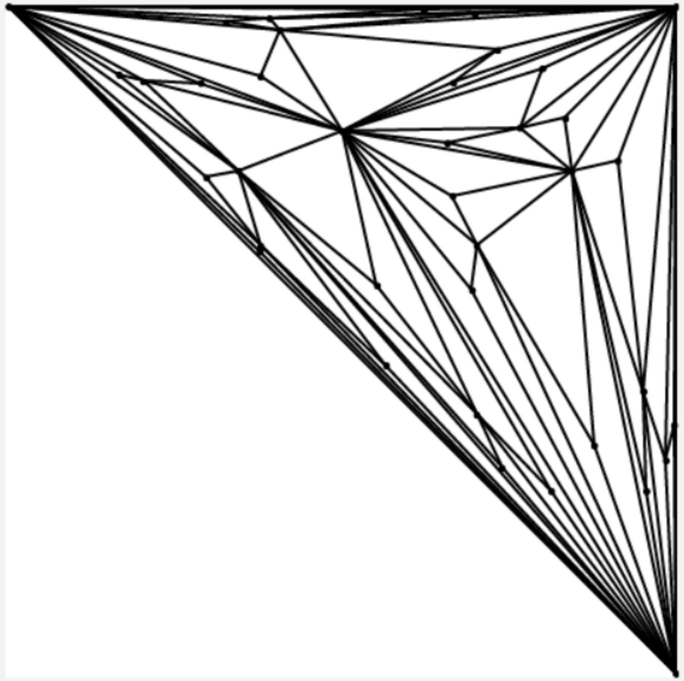


Balanced Random

- We group triangles together
- First choose random group then choose within group



Balanced Random Result



16

DSL

```
shape
    <gtExample>
    | shape |
    shape := SGShapeBuilder new
        points:
            {(#a -> (0 @ 0)).
            (#b -> (1 @ 0)).
            (#c -> (1 @ 1))};
        lines:
            {(#a -> #b).
            (#b -> #c).
            (#c -> #a)};
        build.
    ^ shape
```

Image Generation

```
triangleInlayBalanced
    <script: 'SGImageExamples new triangleInlayBalanced'>
    | builder |
    builder := SGImageBuilder new.
    builder
        from: 1 to: 35 by: 5;
        config: SampleConfigurations new triangleInlayConfig;
        filterIntersections;
        pointColour: Color black;
        pointSize: 5;
        lineColour: Color black;
        lineWidth: 2;
        background: Color white;
        name: 'triangle_inlay_balanced';
        folder: self baseFolder;
        size: 500 @ 500;
        selector: SGBalancedSelector new;
        export
```

What went well

- Subshape detection algorithm works
- Bloc could be used effectively for the editor and the slides
- The Editor helped to find bugs in the interpreter
- The DSL is handy to use

Problems

- I worked on features which werent strictly necessary
- Used Bloc the wrong way (overwrote drawOnSpartaCanvas)
- Better to use composition of Bloc elements

Future work regarding the editor/interpreter

- Improve usability of the editor
- Let the editor catch up with the DSL
- Implement edge cases (1,2 points or straight line)