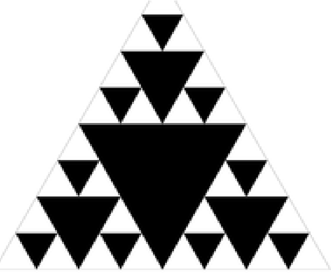
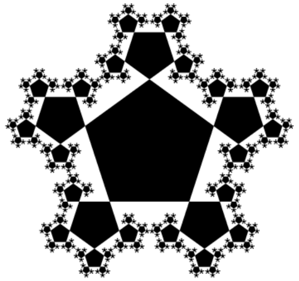
Writing a Shape Grammar Editor

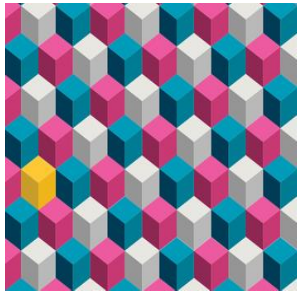
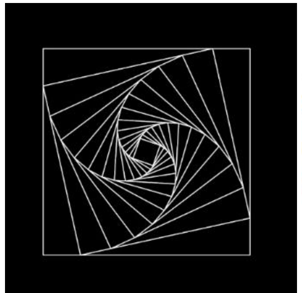
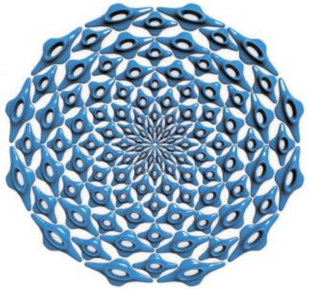
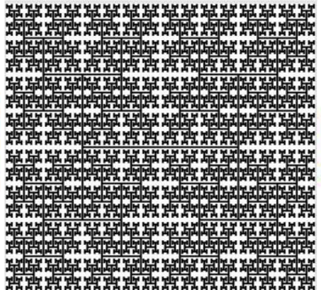
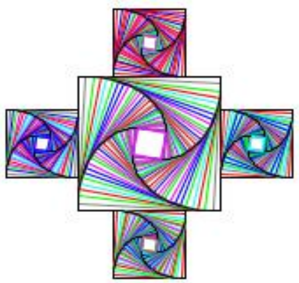
Lars Wüthrich

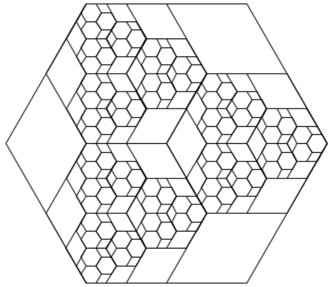
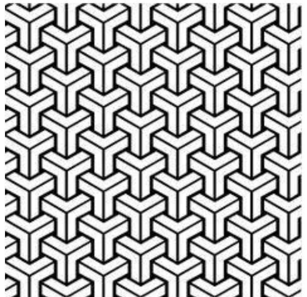
Supervised by Manuel Leuenberger



Bachelor Thesis Fall 2017
Seminar SCG 7th November
SCG University of Bern



What's a Shape Grammar?

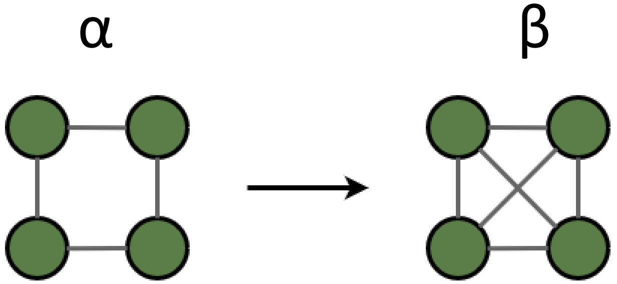
- Defined by George Stiny in 1971

- A shape grammar <S, L, R, I> has four parts:
  1. S, a finite set of **shapes**
  2. L, a finite set of **labels**
  3. R, a finite set of **rules** of the form **α → β**
     where $\alpha \in (S, L)^+$ and $b \in (S, L)^*$
  4. I, a labelled shape $\in (S, L)^+$, called **initial shape**

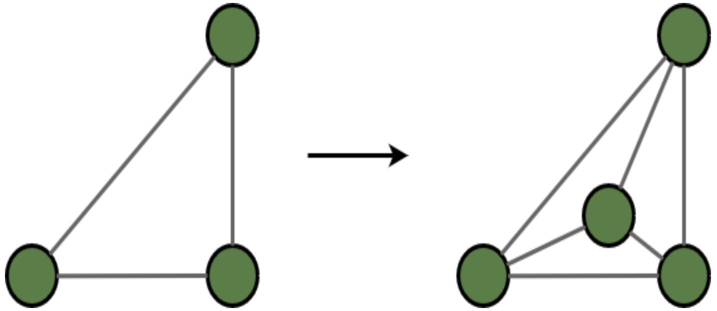
- Creates patterns in 2D, 3D

Rules

- Add edges



- Add points

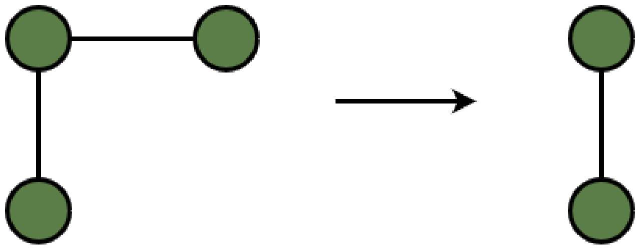


- Remove Edges

- Remove Points

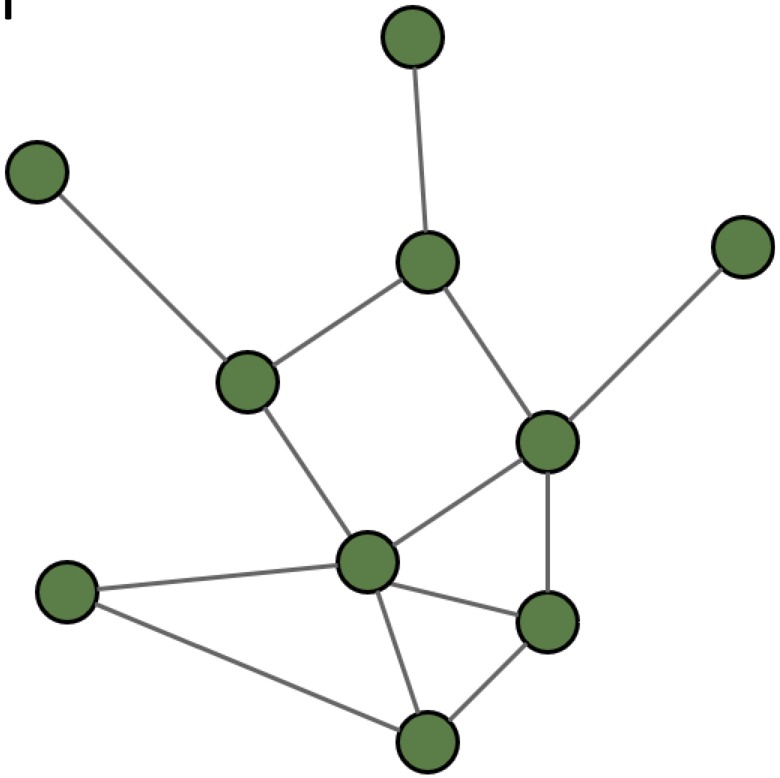
- Scale the shape

- Move points around

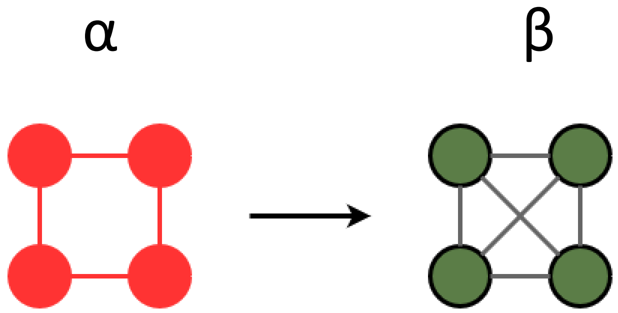


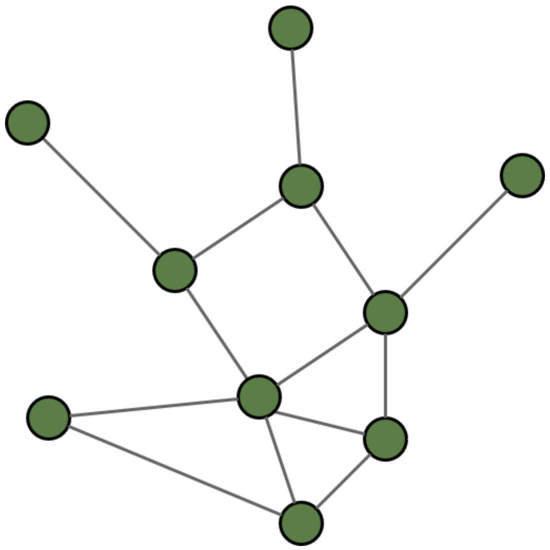
Rule Application

- Start with an initial shape I

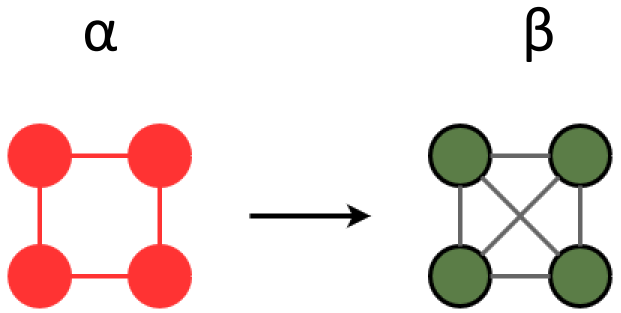


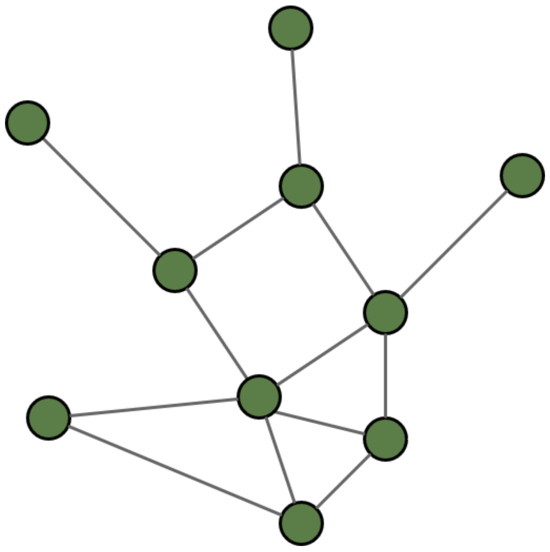
- Find α inside starting shape



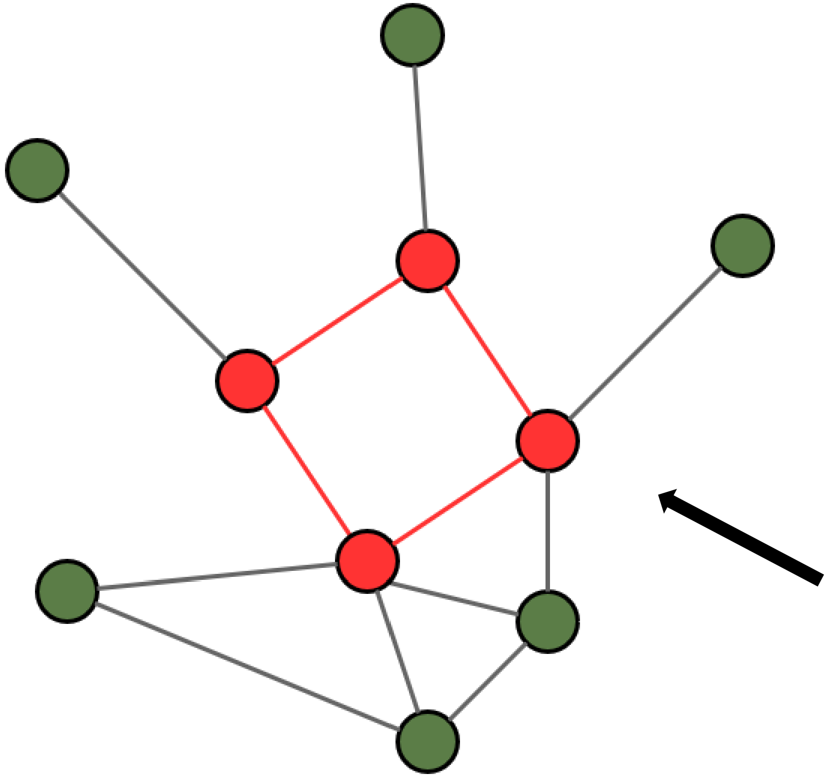


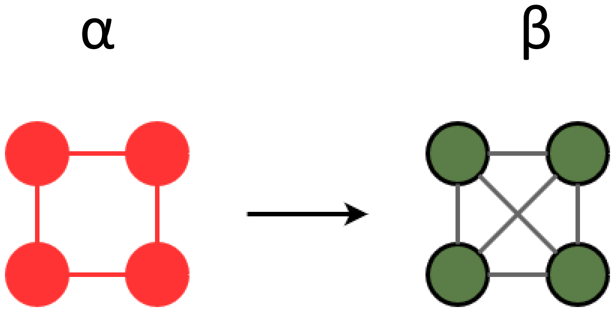
- Find α inside starting shape
- α could be translated, rotated, scaled





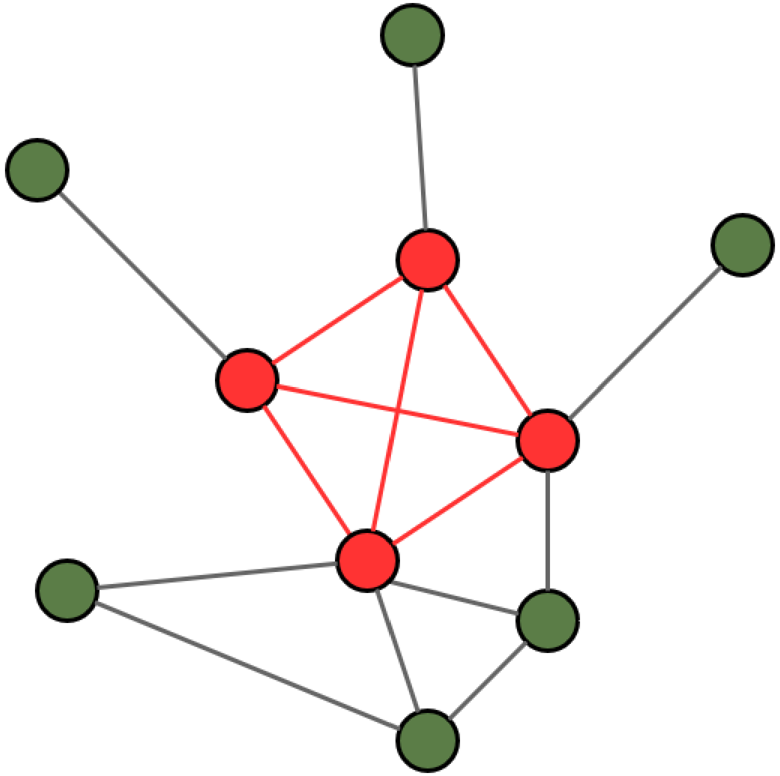
- Subshape Detection

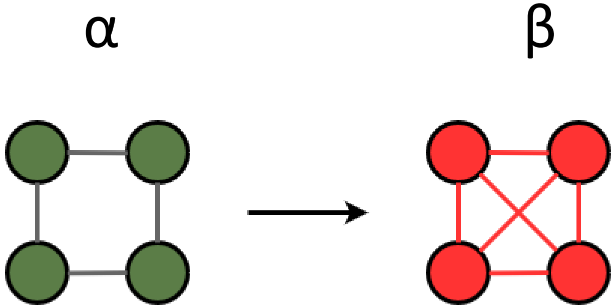




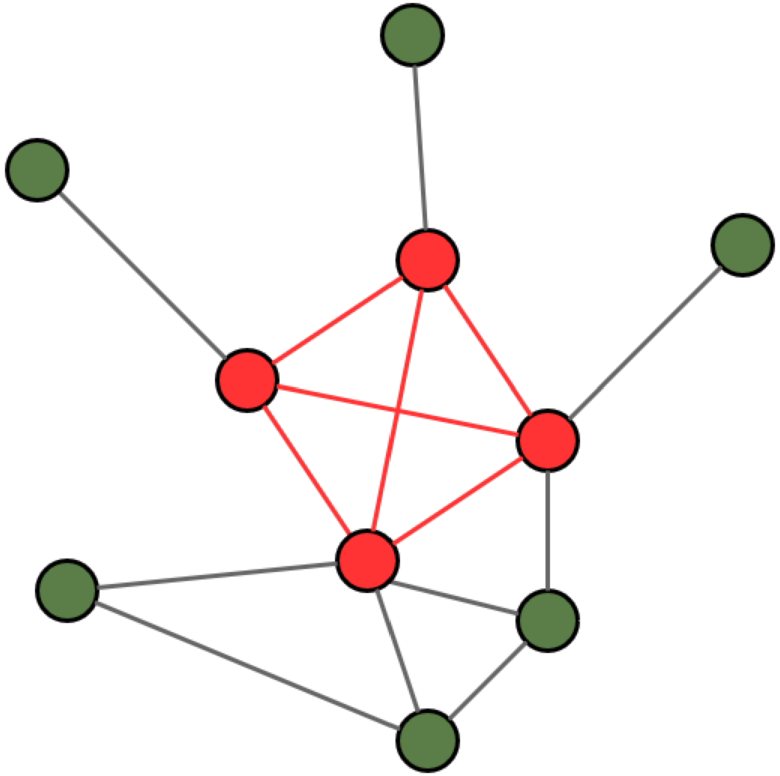
α shape present in
base shape
transformed (rotated)

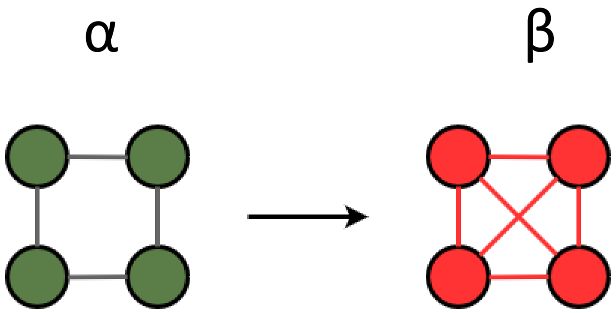
- Apply Rule





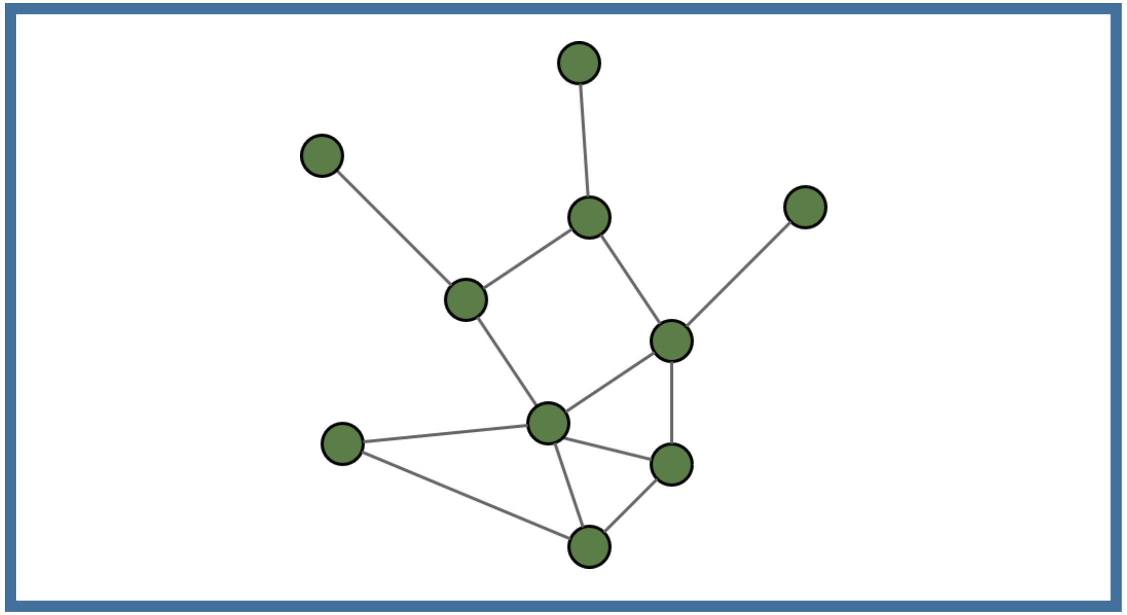
- Apply Rule
- $I' = [I - t(\alpha)] + t(\beta)$



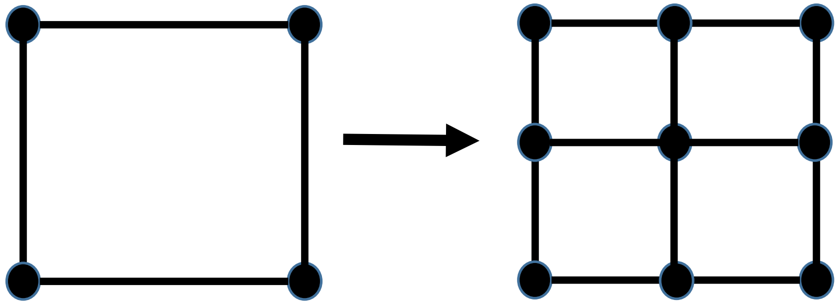


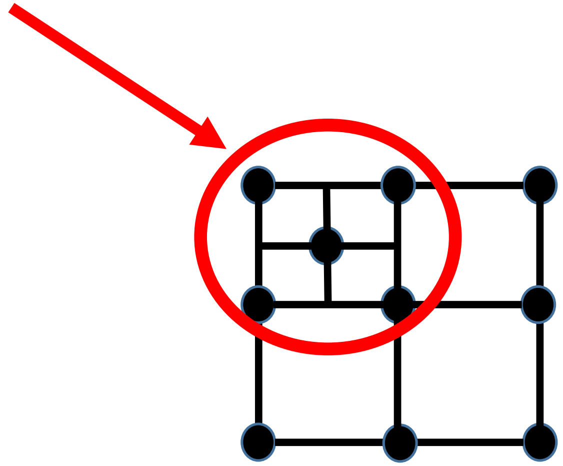
Cases to consider

- Only apply rules in viewport
- Width/Height of desired image



- Apply rules over the whole shape





Not only apply rules here?

- Do not generate geometry below pixel level

- In which order and how often are rules applied?

Labels

- Restrict/Guide rule

- Stop rule application

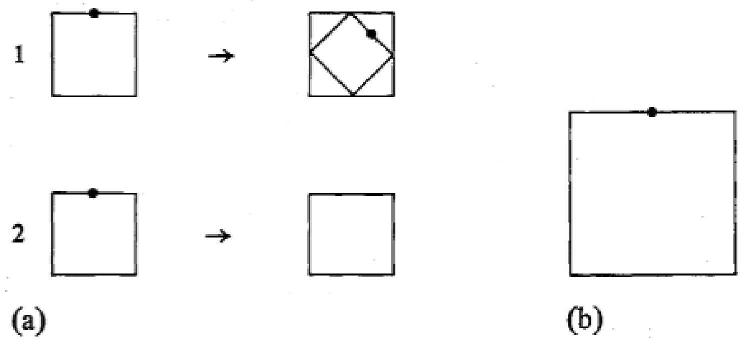
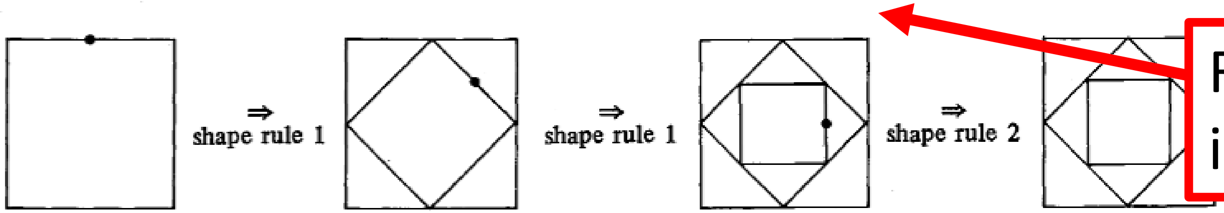


Figure 1. A simple shape grammar that inscribes squares in squares. (a) Shape rules, (b) initial shape.



Rules handdrawn, no implementation

Figure 2. Generation of a shape using the shape grammar of figure 1.

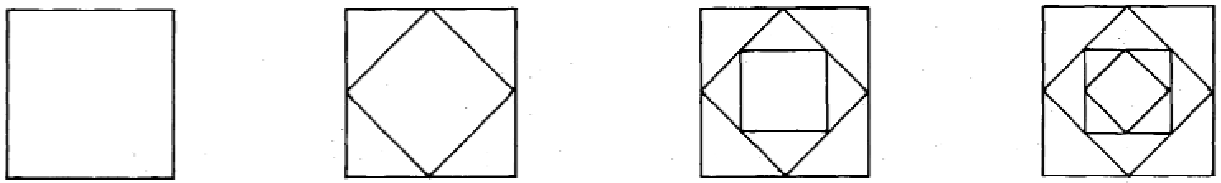
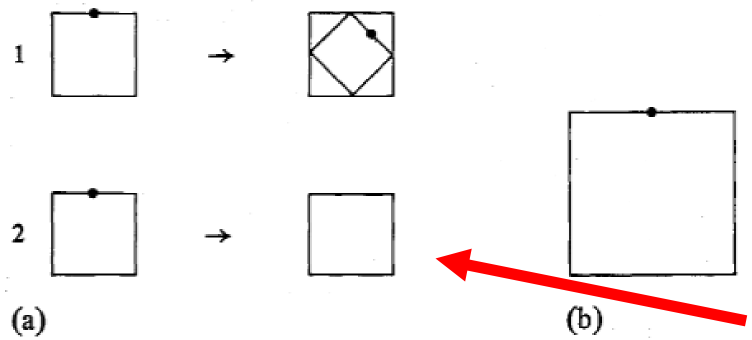


Figure 3. Some shapes in the language defined by the shape grammar of figure 1.

G. Stiny, 1980, Introduction to shape and shape grammars



Rule stops grammar

Figure 1. A simple shape grammar that inscribes squares in squares. (a) Shape rules, (b) initial shape.

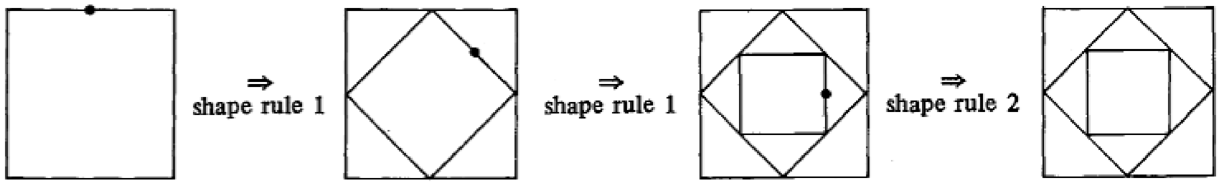


Figure 2. Generation of a shape using the shape grammar of figure 1.

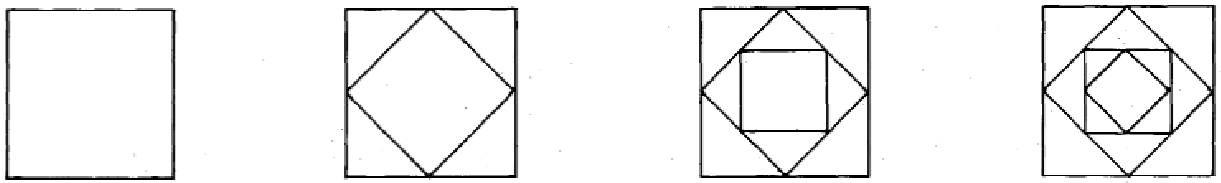
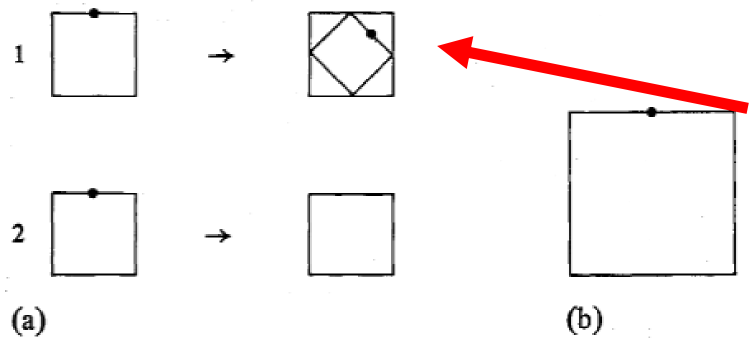


Figure 3. Some shapes in the language defined by the shape grammar of figure 1.

G. Stiny, 1980, Introduction to shape and shape grammars



Label defines orientation

Figure 1. A simple shape grammar that inscribes squares in squares. (a) Shape rules, (b) initial shape.

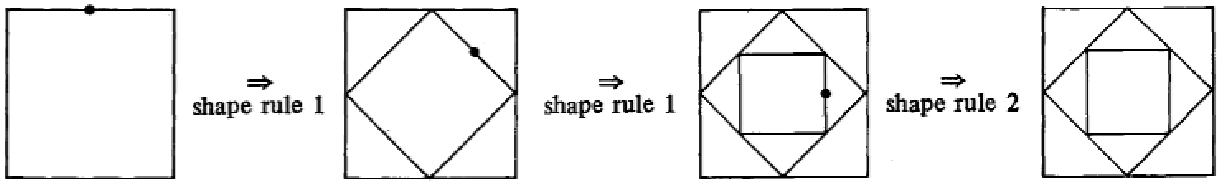


Figure 2. Generation of a shape using the shape grammar of figure 1.

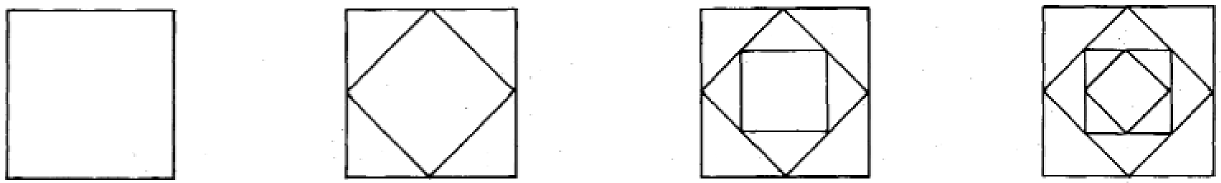


Figure 3. Some shapes in the language defined by the shape grammar of figure 1.

G. Stiny, 1980, Introduction to shape and shape grammars

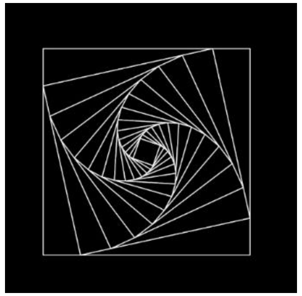
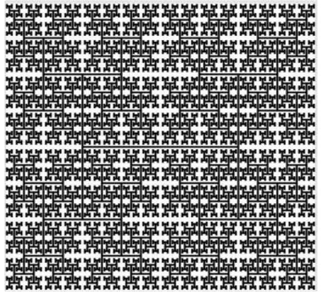
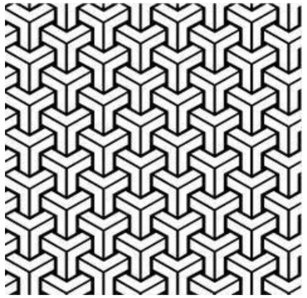
Why do we care?

- Create Textures/Patterns

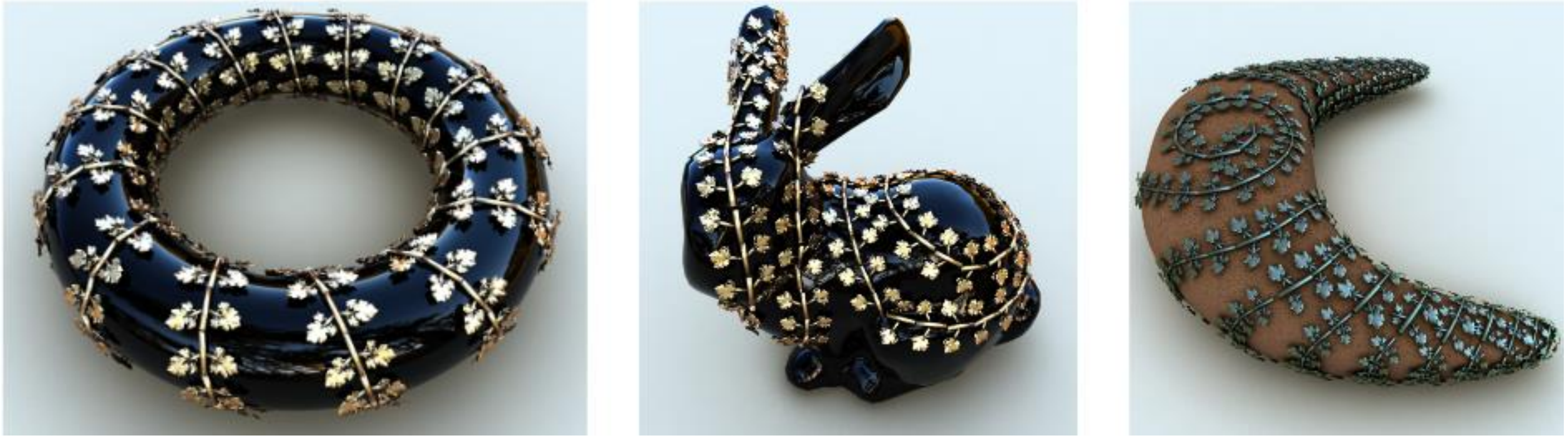
- Create art

- Procedural content for games (room, level design)

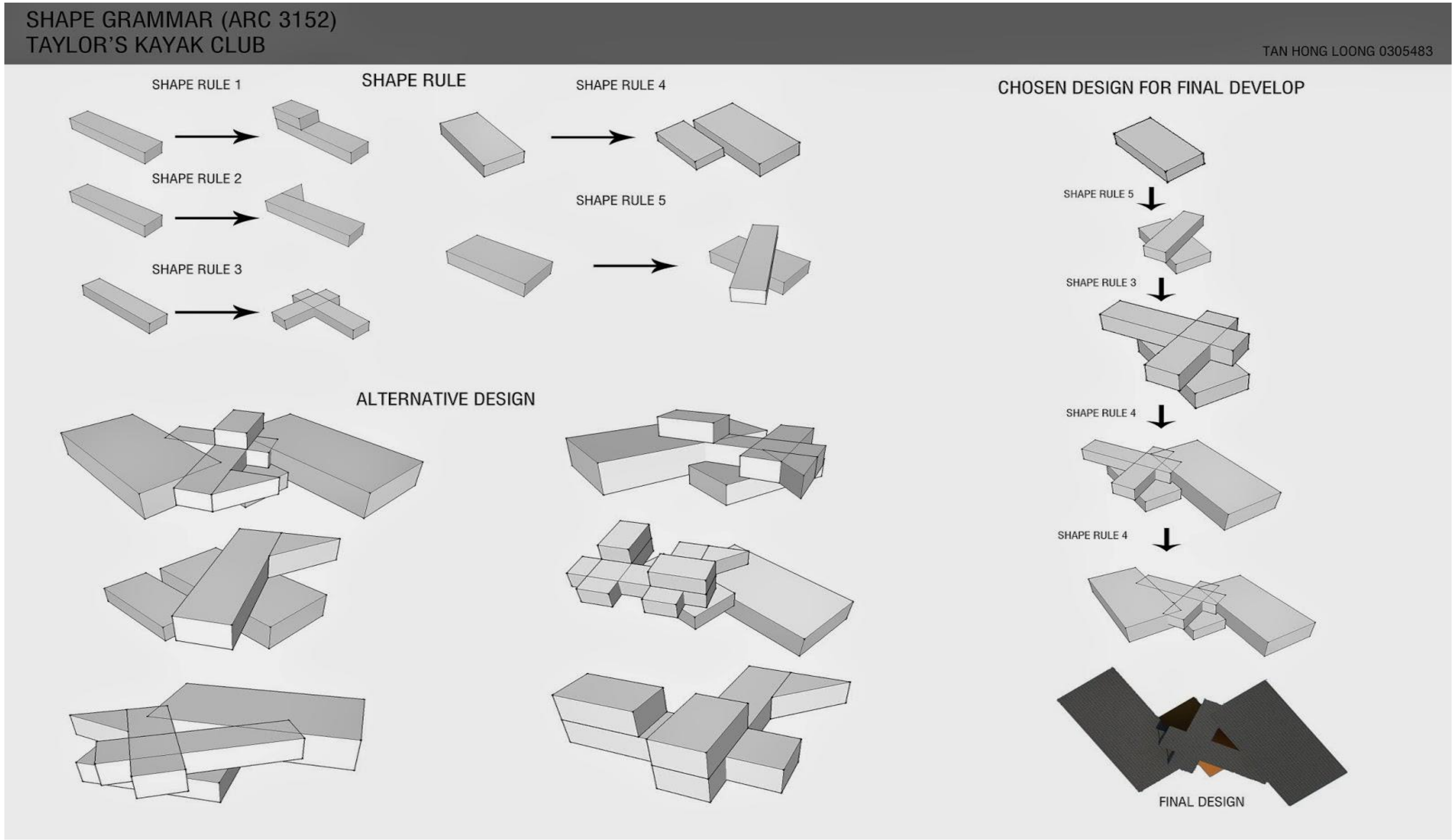
- Tool for designers

- Used in Computer Graphics
- «Geometry Synthesis on Surfaces using Field-Guided Shape Grammars» (2010)



- Use it in architecture in 3D (model generation)

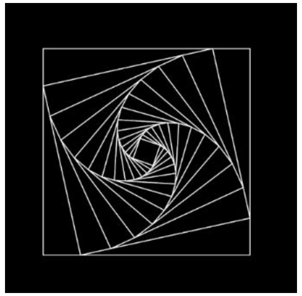
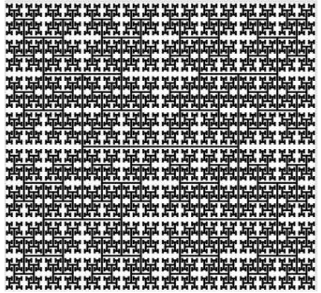
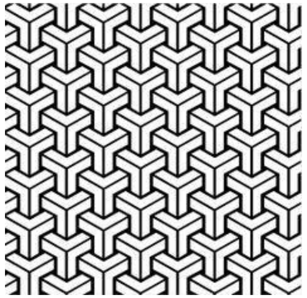


http://tanhongloong-e-portfolio.blogspot.ch/2014/07/shape-grammar-arc3153-arc3152-module.html, 29.10.2017

- Shape Grammars are interesting and cool

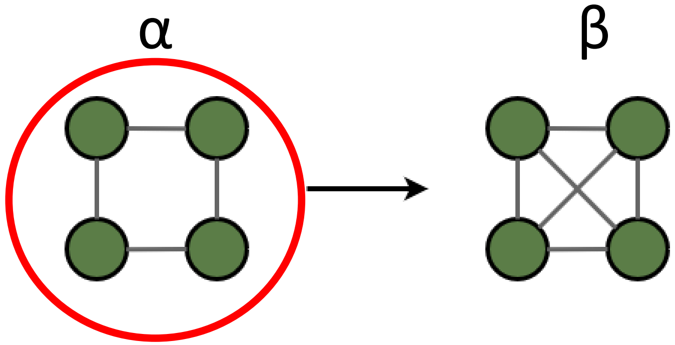
  → visual logic

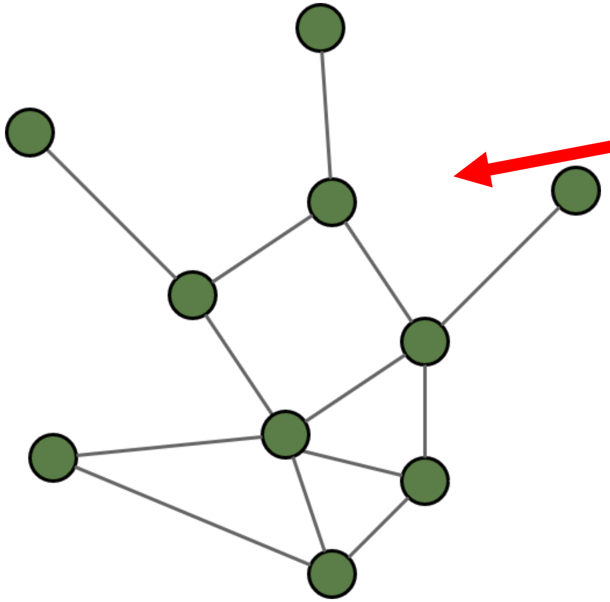
- There has been no unified implementation found yet (?)

Subshape Detection Problem

We need to find this



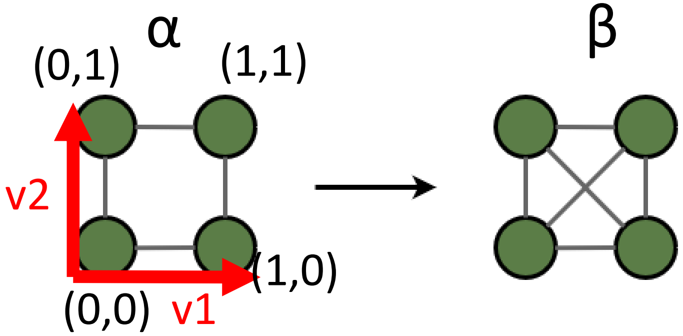


Inside the base shape

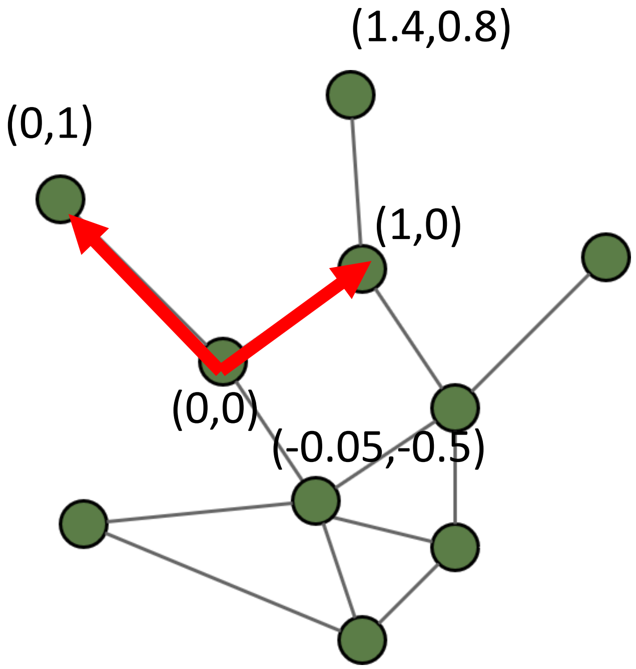
- Rotation, Translation and Scaling can be allowed
- Subshape Detection under these conditions is difficult

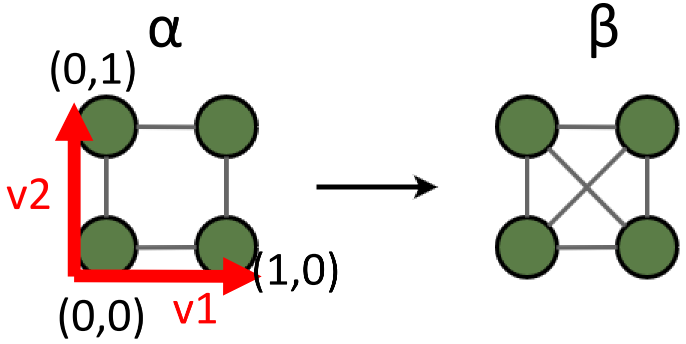
Idea

- Use local coordinates for α

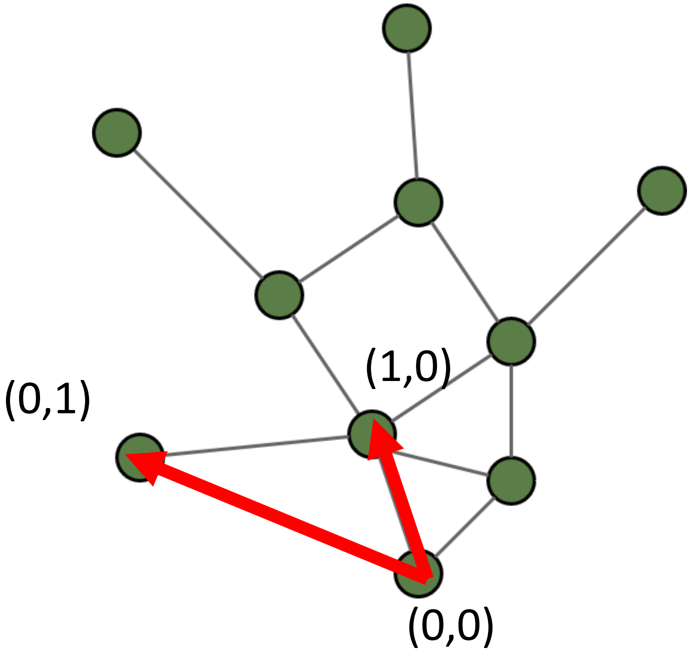


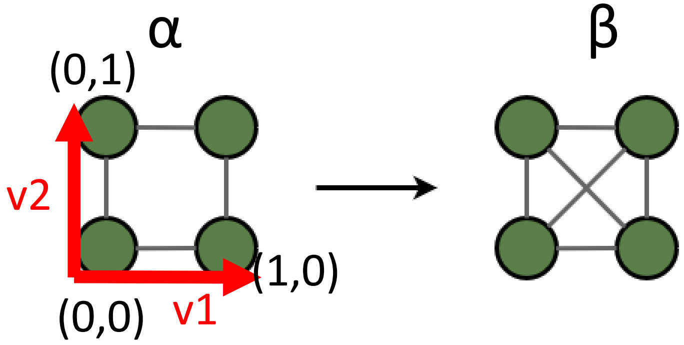
- Local coordinates for every 3 points in base shape
- Compare points



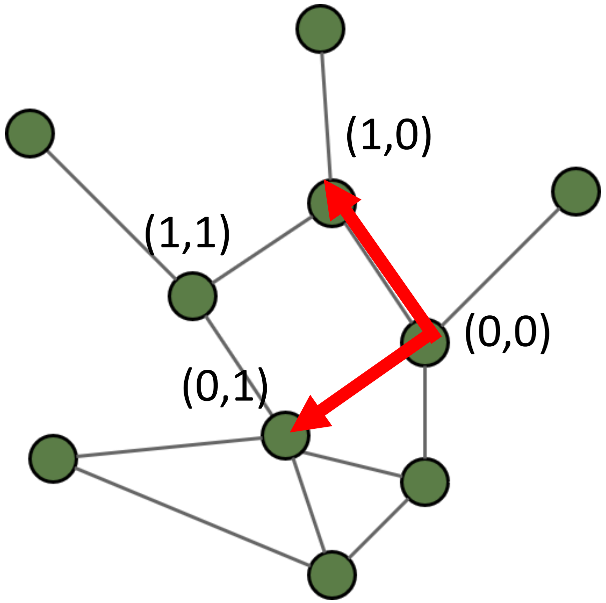


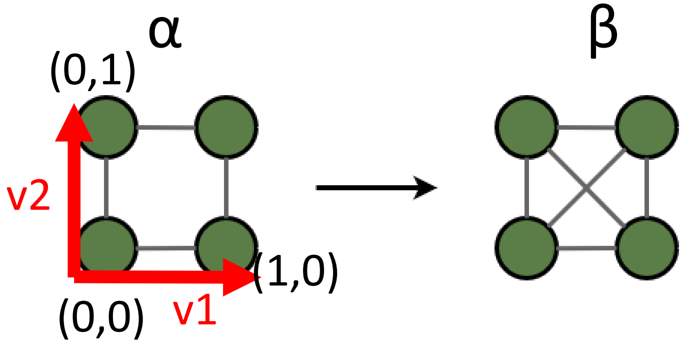
- Local coordinates for every 3 points in base shape
- Compare points



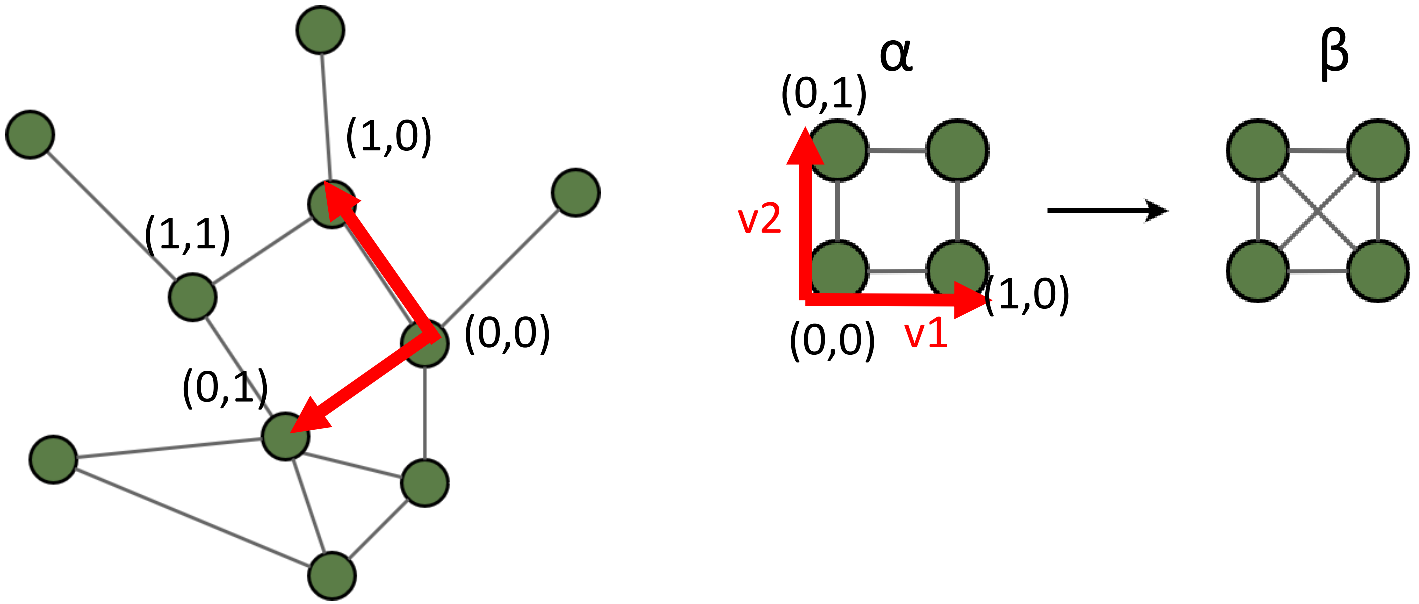


- Local coordinates for every 3 points in base shape
- Compare points



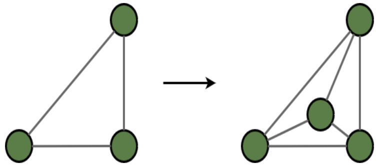


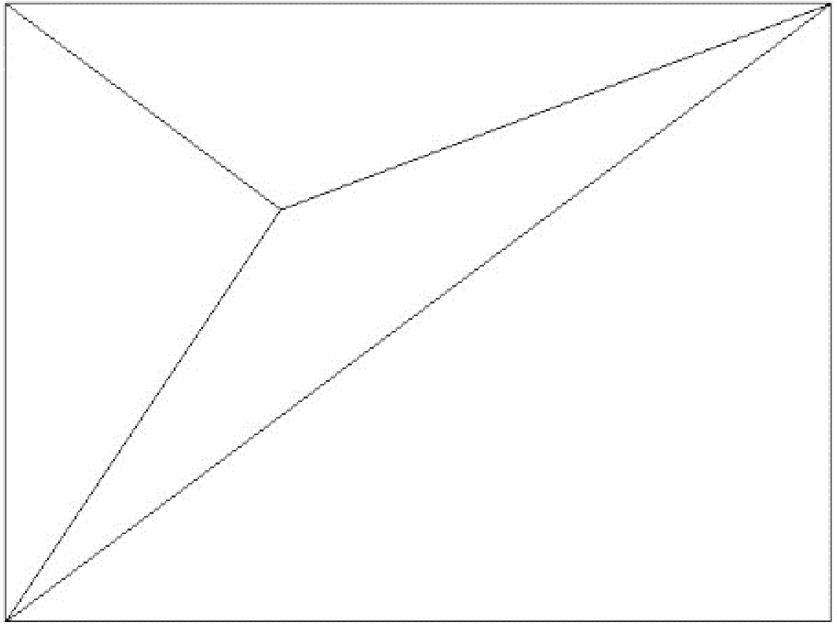
- Local coordinates for every 3 points in base shape
- Compare points
- If match is found compare lines

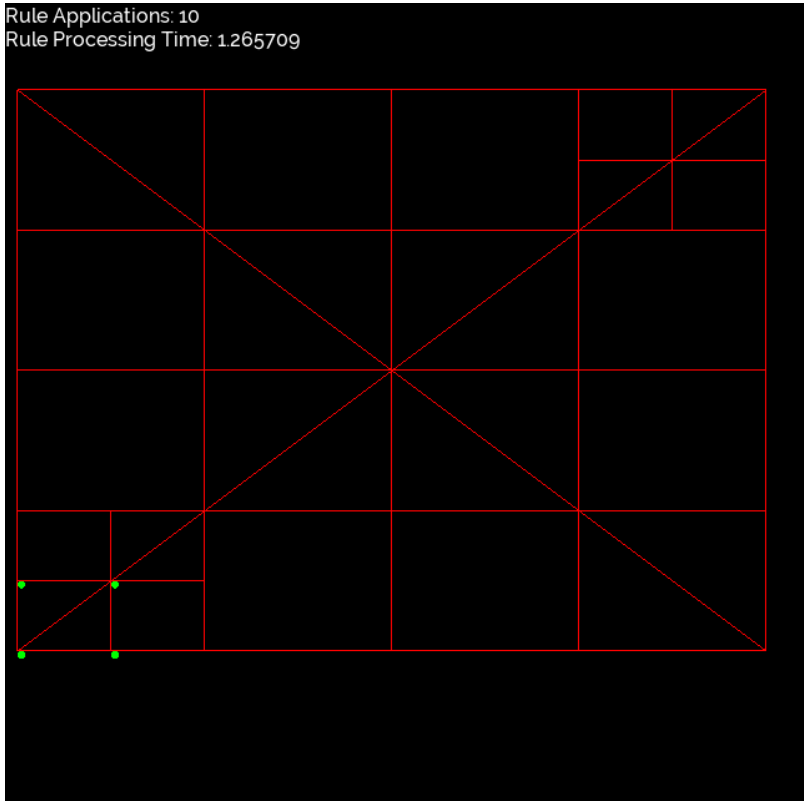


What I have done

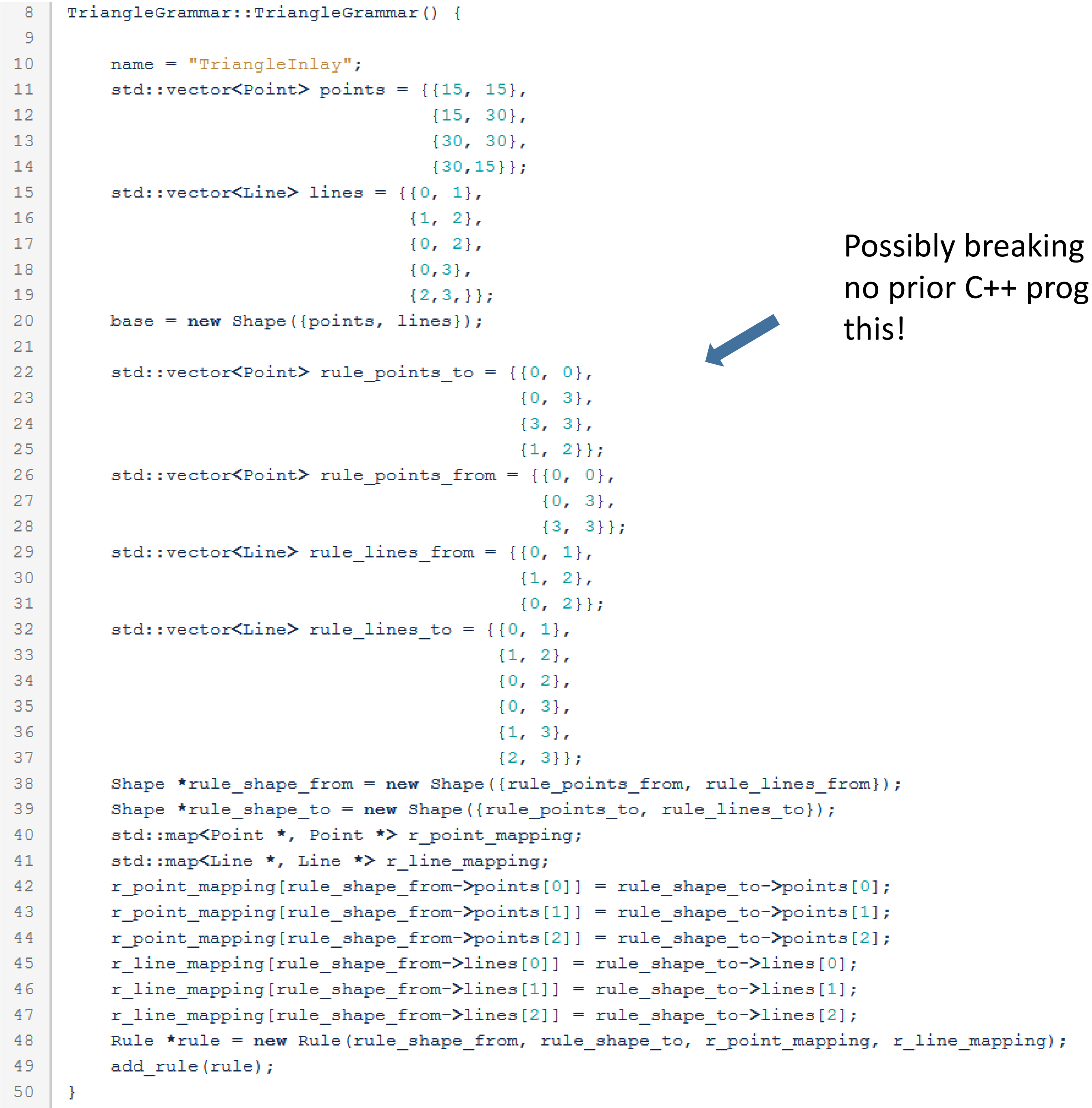
- C++ shape grammar interpreter
- Only for simple grammars
- No editor







Rule Applications: 10
Rule Processing Time: 1.265709

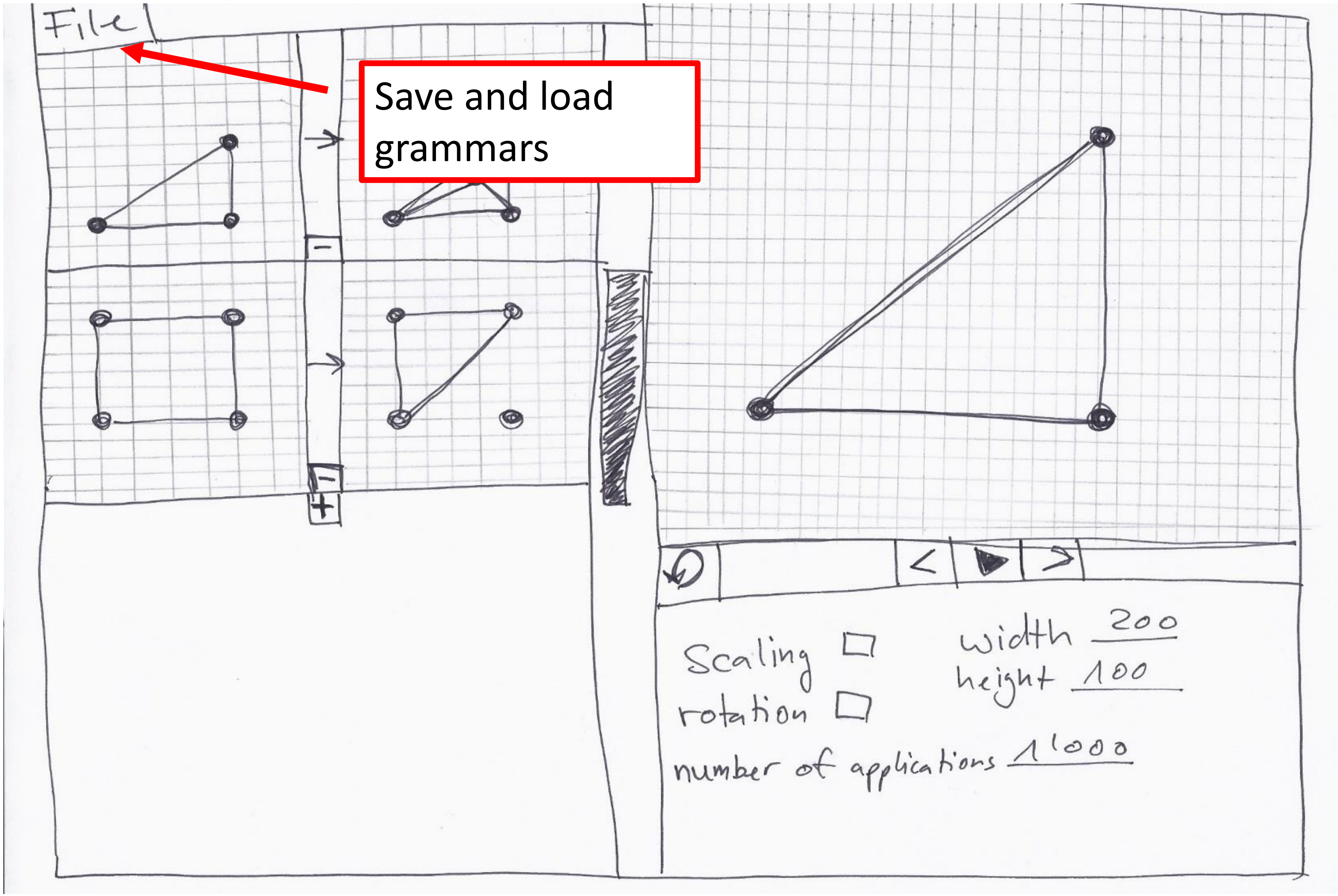


```
TriangleGrammar::TriangleGrammar() {

    name = "TriangleInlay";
    std::vector<Point> points = {{15, 15},
                                  {15, 30},
                                  {30, 30},
                                  {30,15}};
    std::vector<Line> lines = {{0, 1},
                               {1, 2},
                               {0, 2},
                               {0,3},
                               {2,3,}};
    base = new Shape({points, lines});

    std::vector<Point> rule_points_to = {{0, 0},
                                         {0, 3},
                                         {3, 3},
                                         {1, 2}};
    std::vector<Point> rule_points_from = {{0, 0},
                                           {0, 3},
                                           {3, 3}};
    std::vector<Line> rule_lines_from = {{0, 1},
                                         {1, 2},
                                         {0, 2}};
    std::vector<Line> rule_lines_to = {{0, 1},
                                       {1, 2},
                                       {0, 2},
                                       {0, 3},
                                       {1, 3},
                                       {2, 3}};
    Shape *rule_shape_from = new Shape({rule_points_from, rule_lines_from});
    Shape *rule_shape_to = new Shape({rule_points_to, rule_lines_to});
    std::map<Point *, Point *> r_point_mapping;
    std::map<Line *, Line *> r_line_mapping;
    r_point_mapping[rule_shape_from->points[0]] = rule_shape_to->points[0];
    r_point_mapping[rule_shape_from->points[1]] = rule_shape_to->points[1];
    r_point_mapping[rule_shape_from->points[2]] = rule_shape_to->points[2];
    r_line_mapping[rule_shape_from->lines[0]] = rule_shape_to->lines[0];
    r_line_mapping[rule_shape_from->lines[1]] = rule_shape_to->lines[1];
    r_line_mapping[rule_shape_from->lines[2]] = rule_shape_to->lines[2];
    Rule *rule = new Rule(rule_shape_from, rule_shape_to, r_point_mapping, r_line_mapping);
    add_rule(rule);
}
```

Possibly breaking a ton of C++ idioms, no prior C++ programming before this!

My Bachelor Project

- Focus on 2D Shape Grammars
- Implement an editor
- Draw rules
- Draw starting shape



Save and load grammars

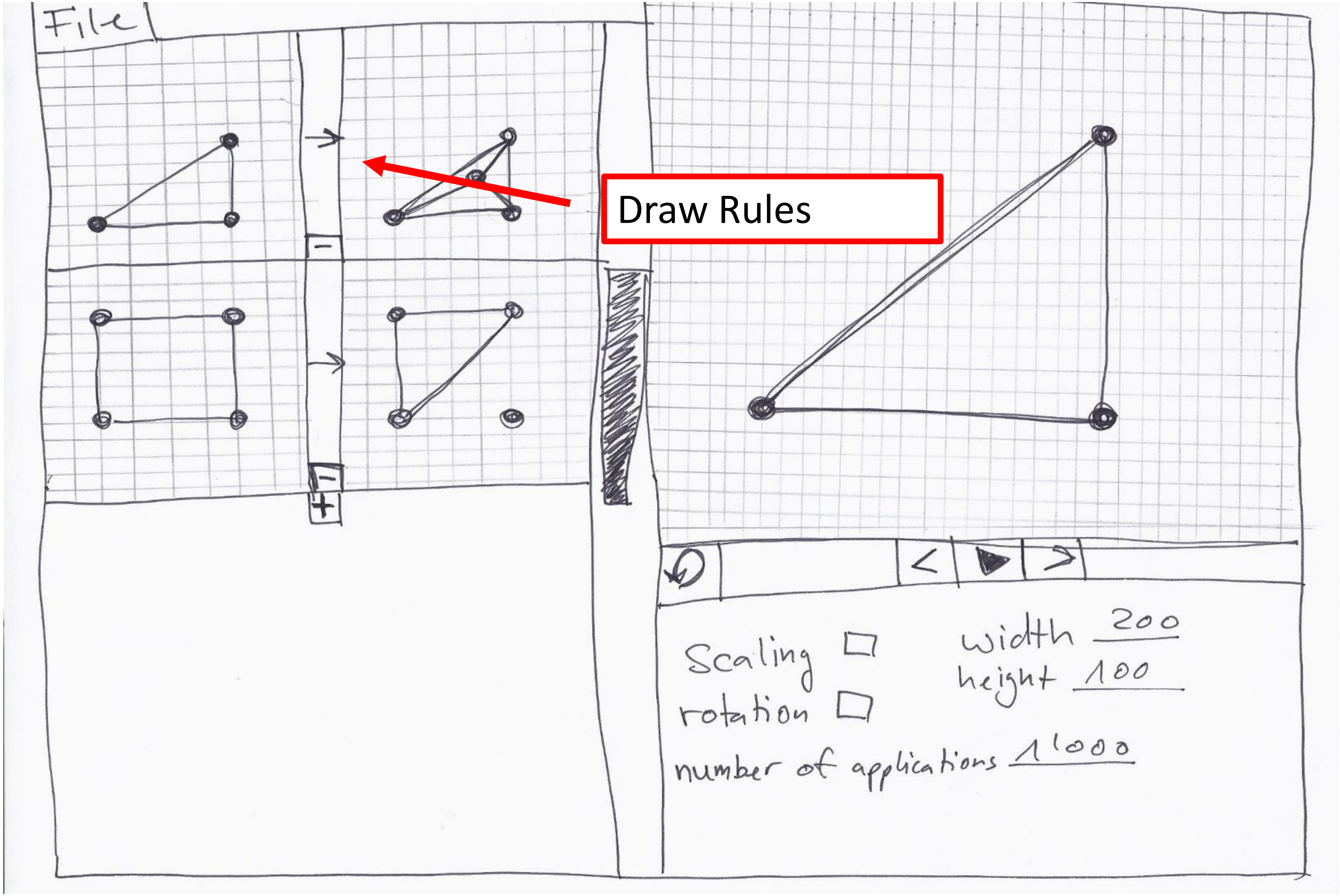
Scaling □

rotation □

width 200

height 100

number of applications 1'000



Draw Rules

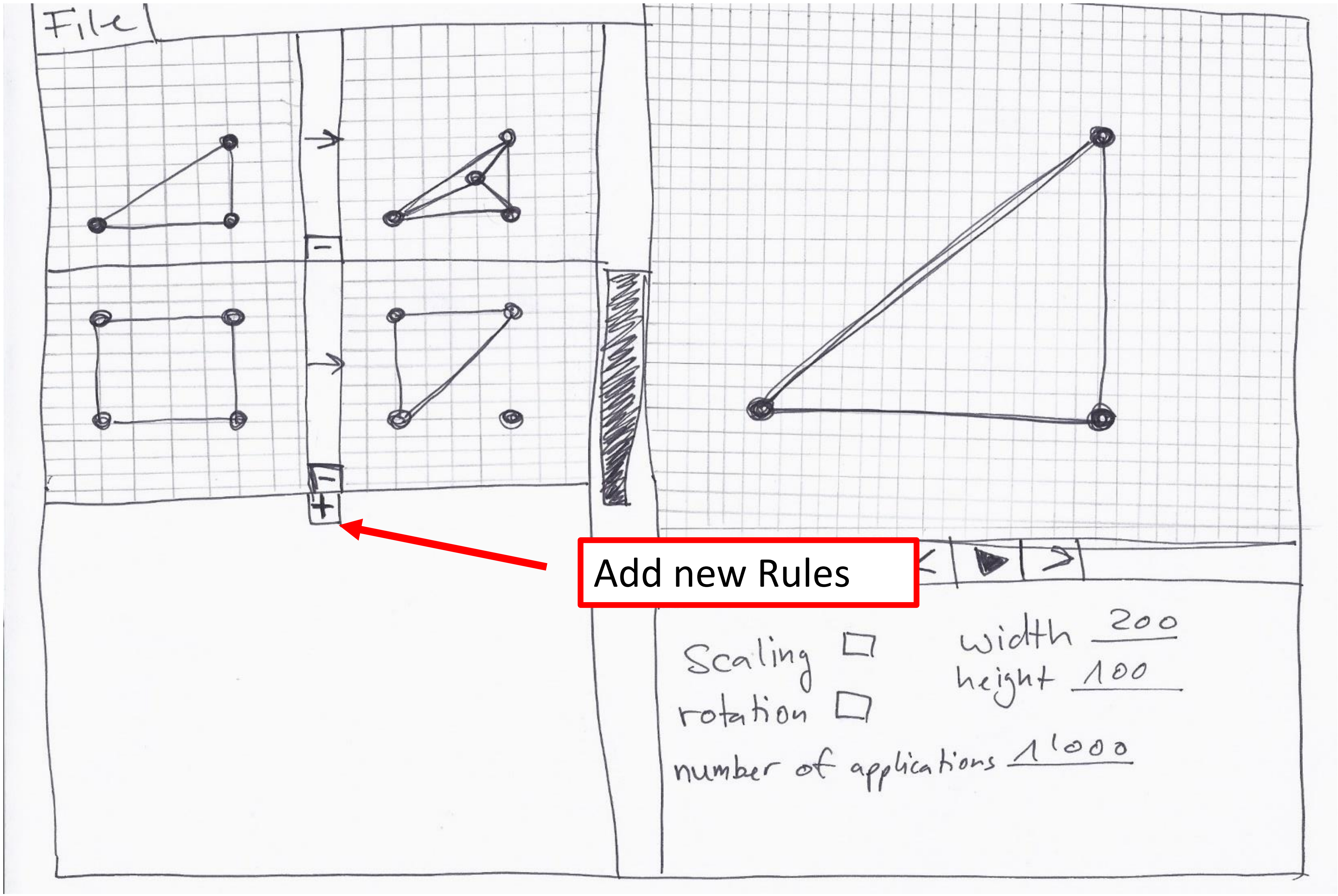
Scaling □

rotation □

width 200

height 100

number of applications 1'000



Add new Rules

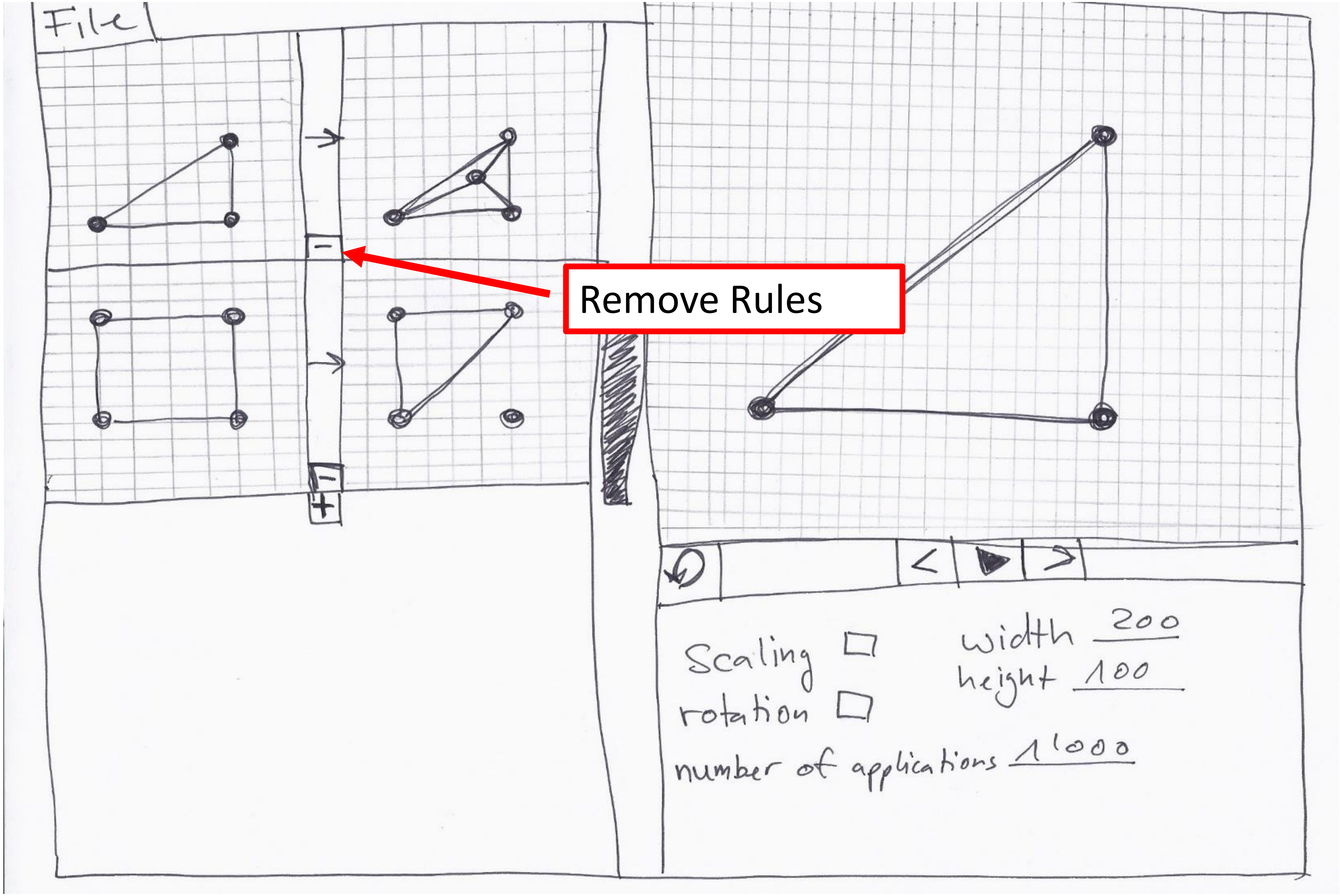
Scaling □

rotation □

width 200

height 100

number of applications 1'000



Remove Rules

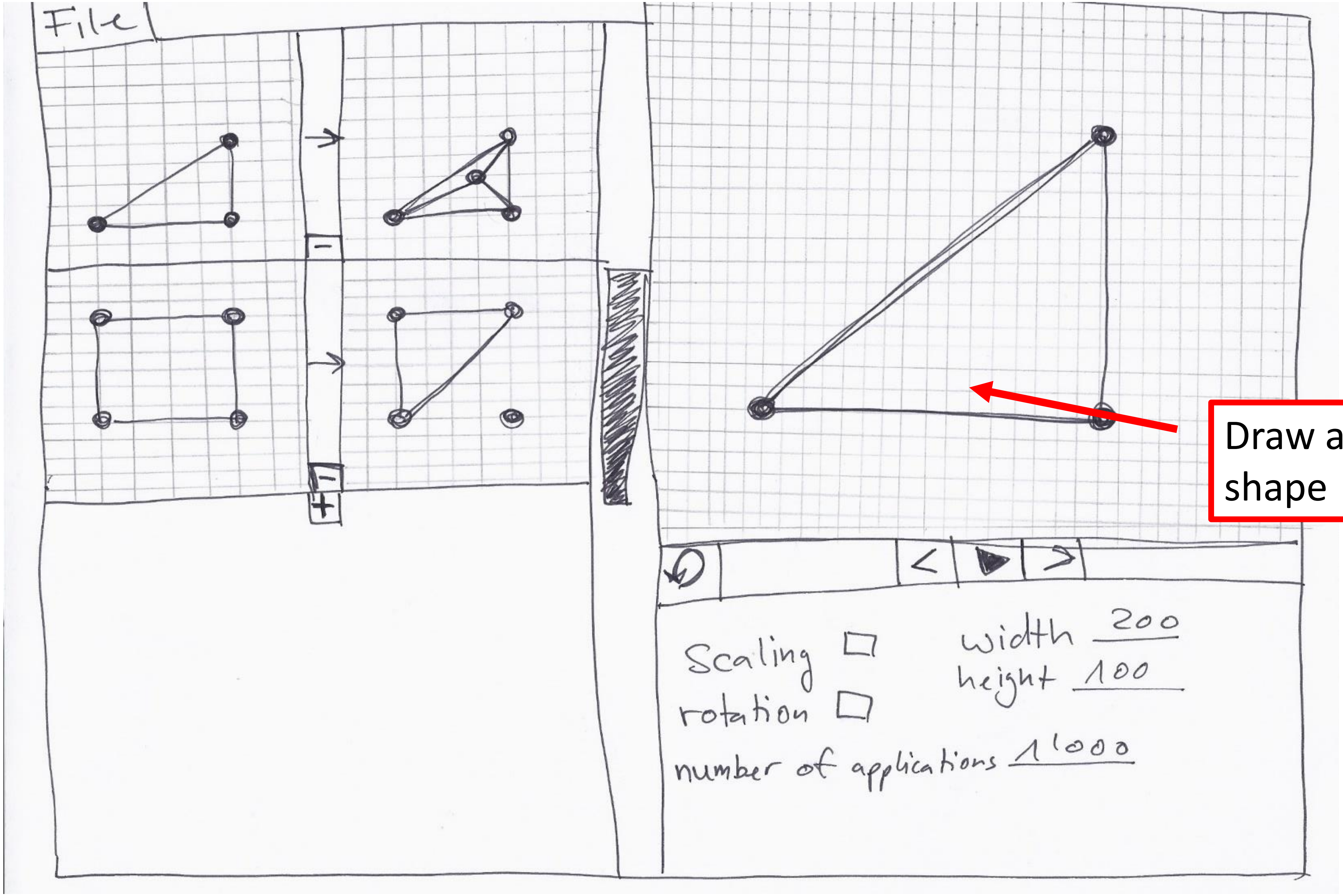
Scaling □

rotation □

width 200

height 100

number of applications 1'000



Scaling □

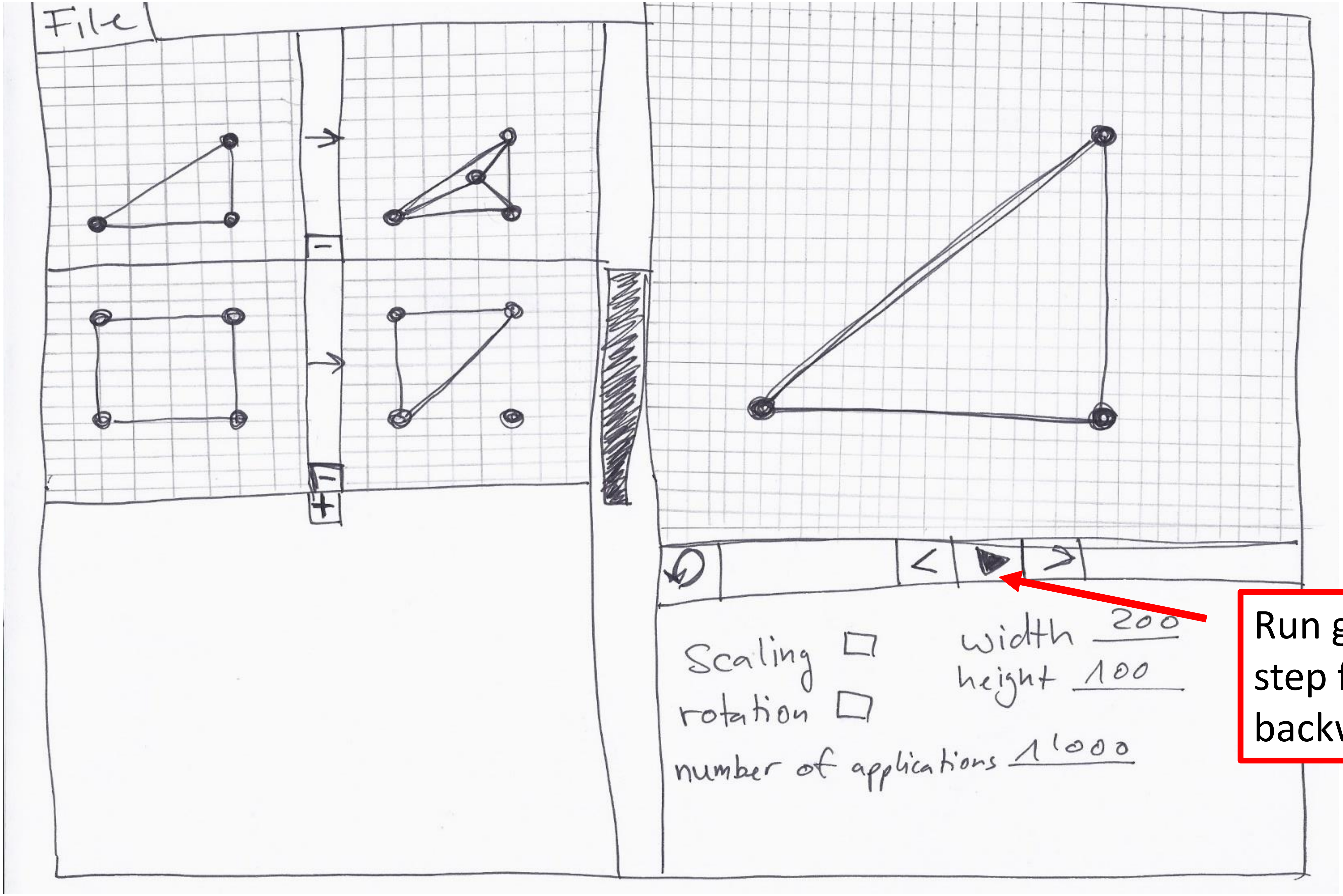
rotation □

width 200

height 100

number of applications 1'000

Draw a starting shape



Scaling □

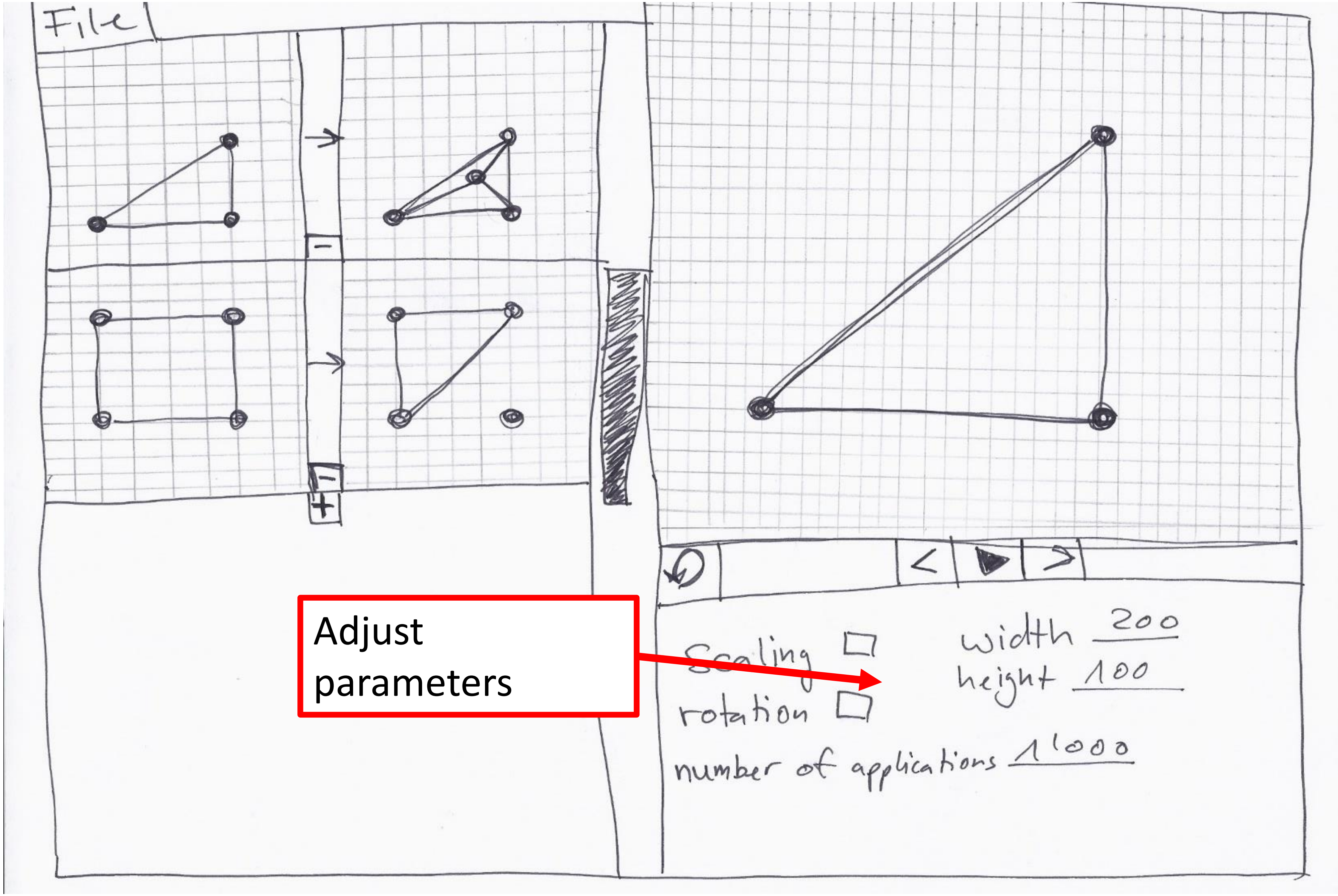
rotation □

width 200

height 100

number of applications 1'000

Run grammar, step forward and backward



Scaling □

rotation □

width 200

height 100

number of applications 1'000

Adjust parameters

Roadmap

- Backend

  subshape detection, shape transformation

- Front end (the editor)

  with spec and roassal or maybe bloc in Pharo

- Test Algorithm

  If it breaks figure out why

- Map software metrics into rules
- Software fingerprint generation
- Add coloring, tagging for further processing

Questions / Feedback

- What do you think about the subshape detection using local coordinates?
- I hope I can create some cool images until the next presentation

References

Image 1:

http://www.elementsofparametricdesign.com/view.php?hash=&dir=files%2FPatterns%2FRecursion, 29.10.2017

Image 2:

From «**Geometry Synthesis on Surfaces Using Field-Guided Shape Grammars”**

https://csdl-images.computer.org/trans/tg/2011/02/figures/ttg20110202315.gif, 29.10.2017

Image 3:

https://introcs.cs.princeton.edu/java/assignments/sierpinski3.png, 29.10.2017

Image 4:

http://www.cs.duke.edu/courses/fall01/cps100/assign/recursivegraph/, 29.10.2017

Image5:

https://i.pinimg.com/originals/24/ca/f7/24caf7f4d101d4fdec36575628f1e319.jpg, 29.10.2017

Image 6:

http://www.cs.princeton.edu/courses/archive/fall08/cos126/art/anya.1.png, 29.10.2017

Image 7:

https://i.pinimg.com/originals/d1/33/77/d1337739ad66deaac7ec57cb018607b8.jpg, 29.10.2017

Image 8:

https://i.pinimg.com/originals/5b/3c/ce/5b3cce3f47c0d0248fa8c98012faaed7.jpg, 29.10.2017

Image 9:

https://i.pinimg.com/736x/a7/d9/c4/a7d9c4129f62712e643536ae30a1106c--islamic-patterns-modern-patterns.jpg, 29.10.2017

Image 10:

https://cdn.dribbble.com/users/1123302/screenshots/2735420/3dpattern_1x.png, 29.10.2017