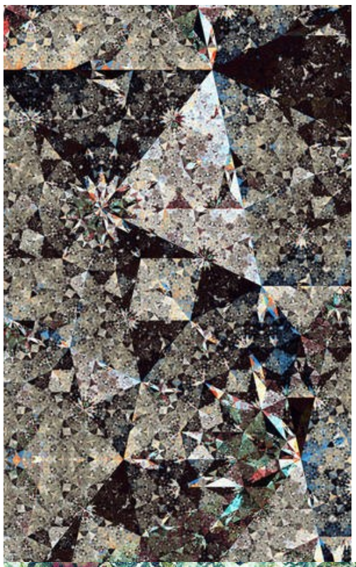
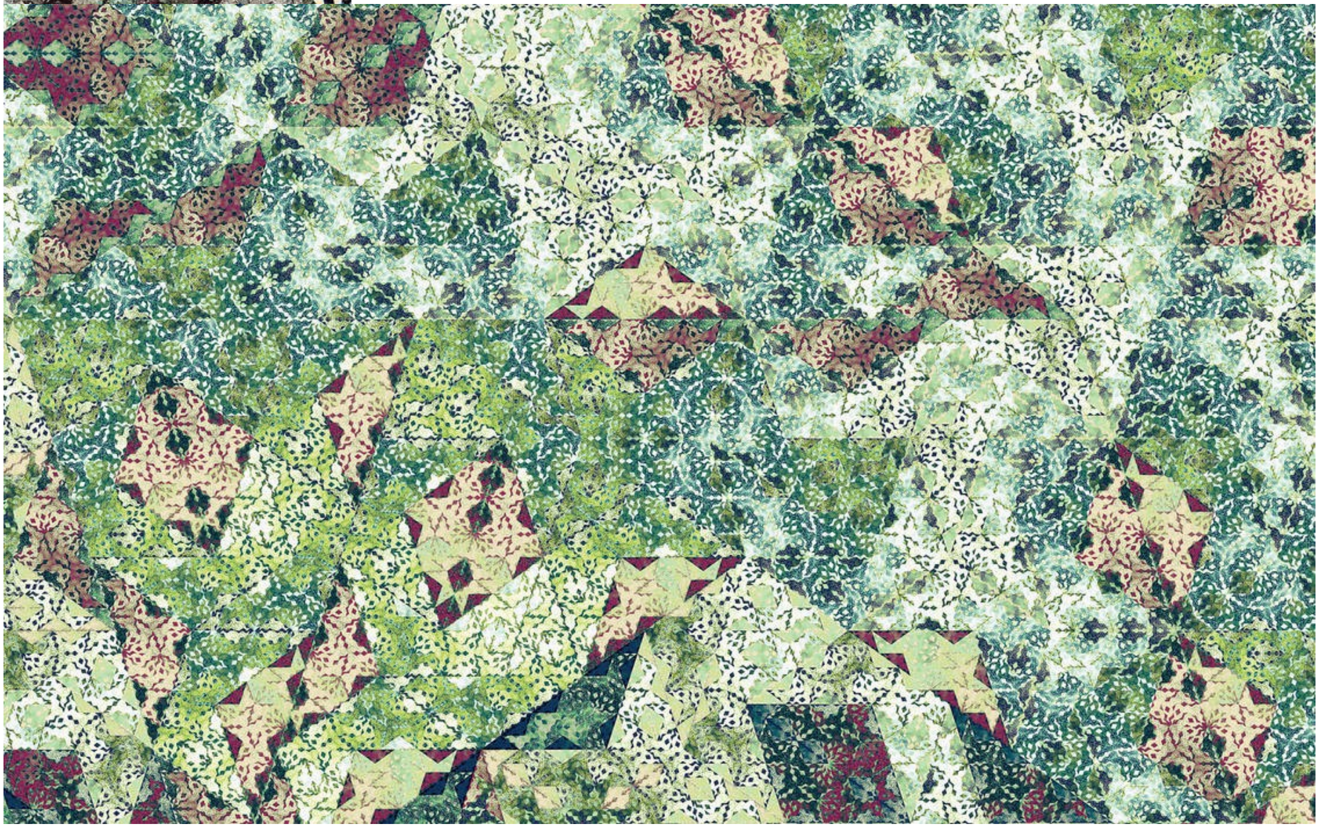
Advanced Linear Tiles by tatasz on DeviantArt

In Linear Tile Tutorial, we learned how to make basic rep-tiles.
Now, what happens if your substitution tiling has pieces of different shapes?

(To implement most of those tiles, you will need some trigonometry knowledge. Life is pain.)





The tiles above are:
tilings.math.uni-bielefeld.de/...
tilings.math.uni-bielefeld.de/...

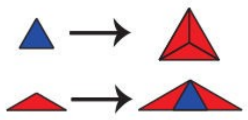
Basic Idea

The tiling principle is the same: each transform corresponds to a tile piece. The transform scale corresponds to the scale of the piece compared to the original shape, and the transform position, to the position of the piece in the tiling.

To allow different shapes, you will need to use xaos - basically, you will use it to specify what pieces are used to make each of the pieces of the tiling.

Some not practical example. Suppose you have the following tile, where the blue triangle is made out of 3 red ones, and the red triangle is made of 2 red triangles and 1 blue triangle.

(from tilings.math.uni-bielefeld.de/)



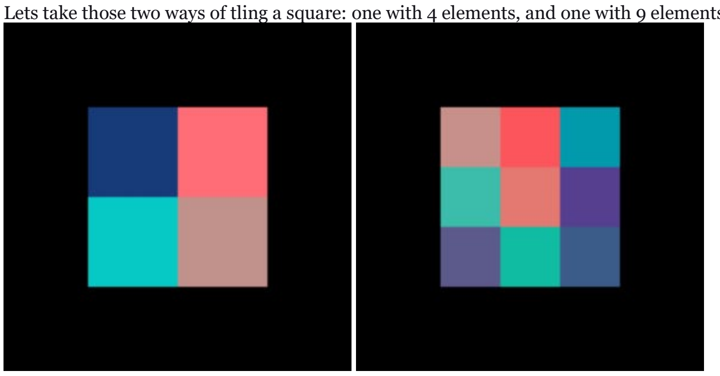
Your fractal will have 6 transforms:

1. a RED triangle as piece of the BLUE triangle
2. a RED triangle as piece of the BLUE triangle
3. a RED triangle as piece of the BLUE triangle
4. a RED triangle as piece of the RED triangle
5. a RED triangle as piece of the RED triangle
6. a BLUE triangle as piece of the RED triangle

Transforms 1 to 3 are pieces of the blue triangle, so they will have "to" weights 1 only to tranforms that are blue triangles - transform 5 only.
Transforms 4 to 6 are pieces of the red triangle, so they will have "to" weights 1 only to transforms that are red triangles - all transforms except 5.
Transform 5 is a blue triangle, so it will have "from" weight 1 only from transforms that are pieces of the blue triangle - transforms 1-3.
Transforms 1 to 4 and 6 are red triangles, so they will have "from" weight 1 only from transforms that are pieces of the red triangle - transforms 4-5.

A very basic example

Lets take those two ways of tling a square: one with 4 elements, and one with 9 elements:



Try making those 2 squares, with side length 2.

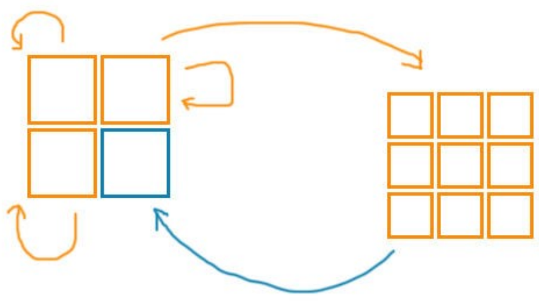
Walkthrough of the 2x2 square:

- Take origin (0,0) as the center of the square.
- Then, its clear that the side of each of the 4 smaller squares should be equal to 1
- So, the scale should be 50% of the original
- And their centers should be respectively at (0.5, 0.5), (-0.5, 0.5), (-0.5, -0.5) and (0.5, -0.5)

Try figuring out the 3x3 square yourself.
If you fail, here are the parameters: [3 x 3 square params]

When you make lets say a 2x2 square, in your fractal, the main square will be tiled with 4 smaller squares. Each of those, will be also tiled wth 4 smaller squares, and so on.

Now, what if we want a 2x2 square, but made in a way that one of the tiles will be a 3x3 square made of 2x2 squares? Too crazy, no, lets draw a fluxogram of this:

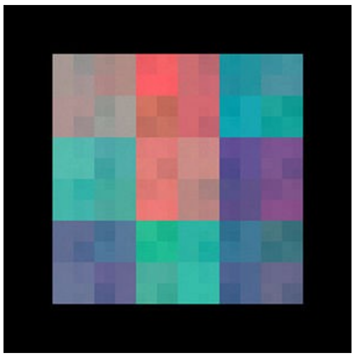
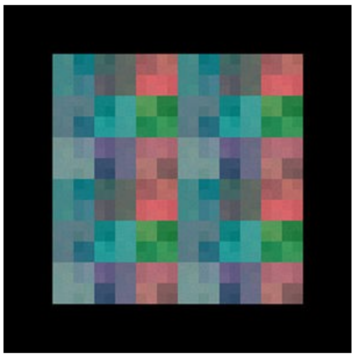


The orange tiles are filled in 2x2 pattern, and the blue one is filled in 3x3 pattern.

Well, lets just take each square on the diagram above as a transform, and that does the trick.

Step 1: without bothering with weights or whatever, first place 9 transforms for the 3x3 square, and then 4 transforms for the 2x2 square, both centered at (0, 0) and with side 2.
Step 2: set the opacities of transforms 1 to 9 to 0 (the 3x3 square).

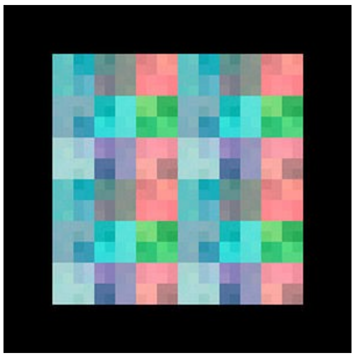
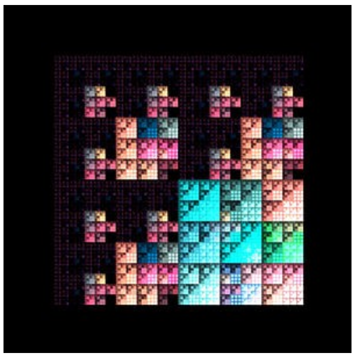
Step 1 (left) and Step 2 (right)

Step 3: remember, transforms 1 to 9 tile into the 3x3 square, and transforms 10-13, into the 2x2 square. According to fluxogram above, the elements of the 3x3 square are made of 2x2 squares, so they should only recieve points from the transforms corresponding to the 2x2 square. So, for transforms 1 to 9, set all from xaos values to 0, except for ones from transforms 10-13.

Step 4: now, 3 of the 4 tiles of the 2x2 square are also 2x2 squares. Repeat the step 3 for the corresponding transforms.

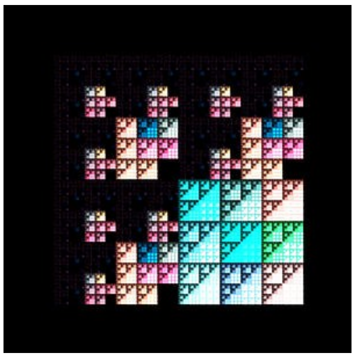
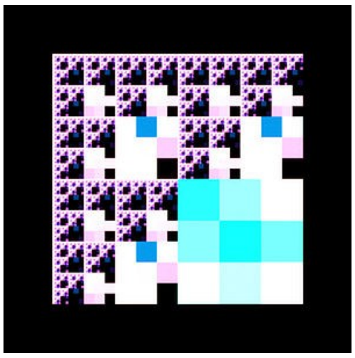
Step 3 (left) and Step 4 (right):

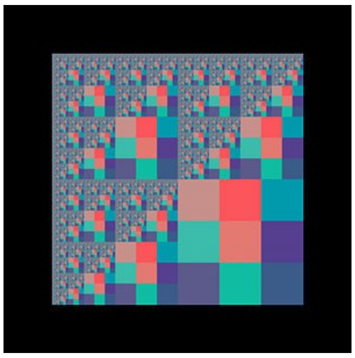
Step 5: we want the last piece of the 2x2 square to be a 3x3 square. Set its from weights to 0, except theweights from transforms 1 to 9, which correspond to the 3x3 square.

Step 6: equalize weights - set the weights of transforms modified in step 4 to 4.5 instead of default 0.5.

Step 5 (left) and step 6 (right)

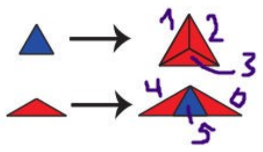



Below, I tweaked the colors a bit to display the structure more clearly.
Also, the final params are avaliable here: 3x3 and 2x2 squares



Equthirds

The example in the beginning is actually the following tiling: tilings.math.uni-bielefeld.de/....
Lets implement it.



(The numbers below are pure satanic magic and require basic trigonometry only)

The sale factor is 0.57735 (sqrt(3) / 3), as you can verify.

Taking as origin the left corner of both triangles, we have the following transforms (all scaled down to 0.57735 of the original size):

1. No shift, flip vertical and rotate 60 CCW
2. Shift 1 unit to the right and rotate 120 CCW
3. No shift, no rotation
4. No shift, flip vertical and rotate 30 CCW
5. Shift 0.57735 unit to the right, no rotation
6. Shift 1.73205 unit to the right and rotate 150 CCW

Now, set weights as described in the beginning of this tutorial.

The weight normalization factor here is 3, so set the weights of transforms 4 and 6 to 1.5 (0.5 * 3).

Finally, set the opacities of transforms 1 to 3 to 0, to avoid overlaps: Equithirds