Fractals from Hinged Hexagon and Triangle Tilings: Supplement

H. A. Verrill

Warwick University, UK; H.A.Verrill@warwick.ac.uk

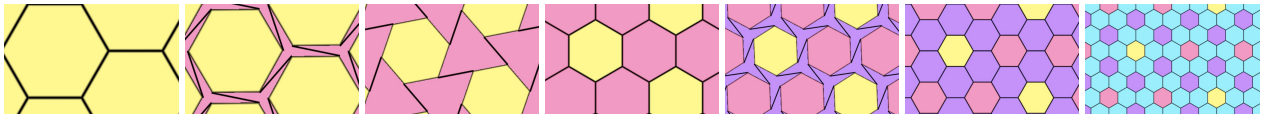
Abstract

This is a supplement to the *Fractals from Hinged Hexagon and Triangle Tilings* submission to the Bridges 2024 proceedings [7]. Here larger images are shown, and further details of the constructions are given.

There are three kinds of fractal procedures described in this supplement. For each type, we have the fractal from the foreground curves on the tiles, and also the pattern created by the background tiles, either coloured with one colour per tile, or via the background colouring algorithm described in the Bridges paper [7]. Note that some images in this paper have lower quality in order to reduce file size. The original images can be reproduced to the desired quality using the program [8].

1 Hexagon Tilings

1.1 Background Tiles



(a) $t = 0$ **(b)** $t = 0.1$ **(c)** $t = 0.5$ **(d)** $t = 1$ **(e)** $t = 1.2$ **(f)** $t = 2$ **(g)** $t = 3$

Figure 1: *Hinged hexagon tiling with links. Three complete applications of hinging operation* 0 *shown.*

The hinged hexagon tiling in the first section of [7], inspired by the description at [2] can be continuously varied using the program found at [8]. A few of the stages are shown in Figure 1. Notice that in this figure, each newly introduced background colour takes up twice the space of the union of all previous colours, e.g.,

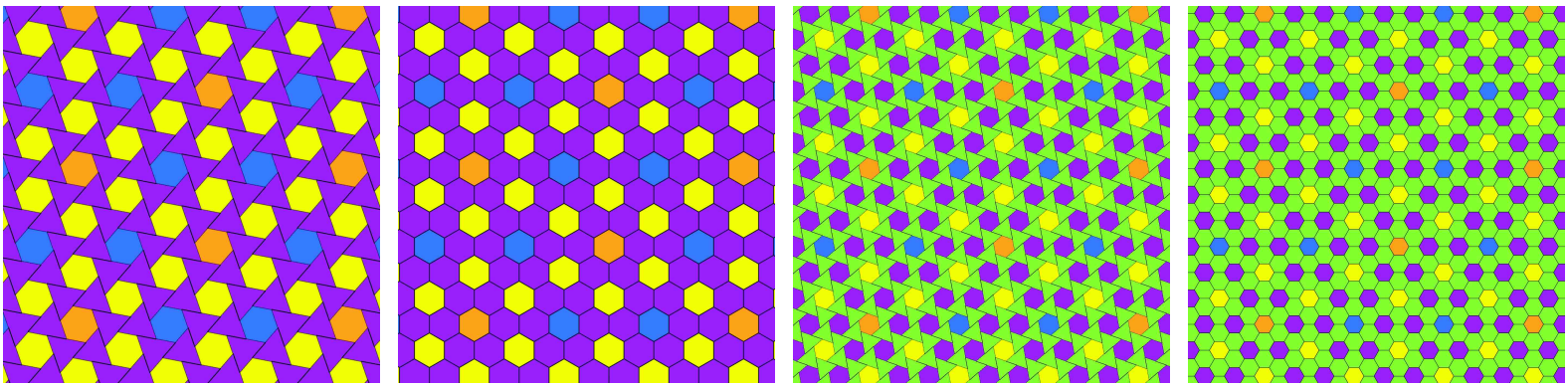


Figure 2: *Continuation of Figure 4 in [7] (scaled up), for the operation sequence* 0110*, applied to stages* $t = 3.5, 3, 3.5, 4$ *times respectively*

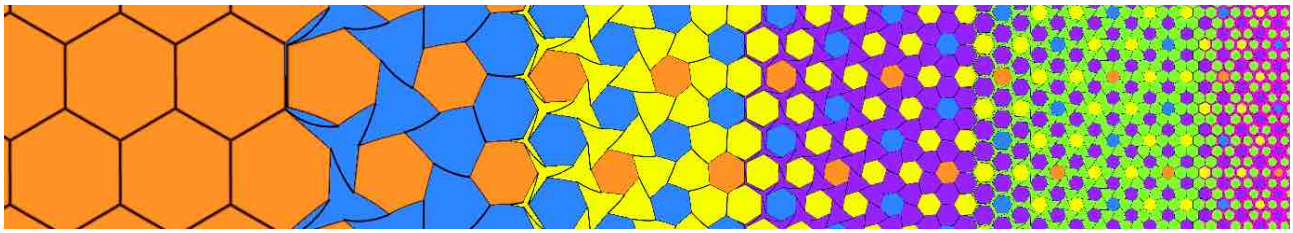


Figure 3: *Variable iteration level*

in (g), there are twice as many cyan tiles as the other colours combined. This means if we vary the iteration level continuously, we will see clear bands of different colours, as in Figure 3. The iteration level can be varied as desired across the image, e.g., some circular gradient options are given in Figure 4 In some of

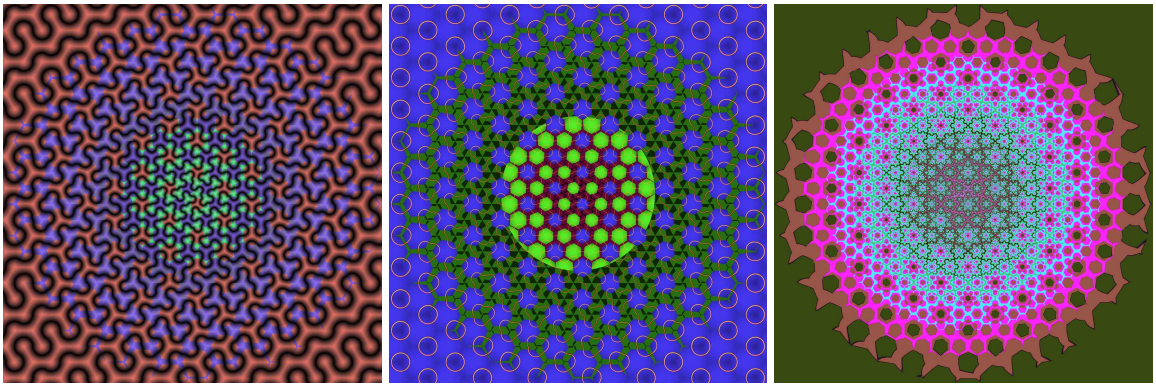


Figure 4: *Variable iteration level from centre*

the variable iteration images, there is a sharp jump at certain places. This happens if aspects of the image

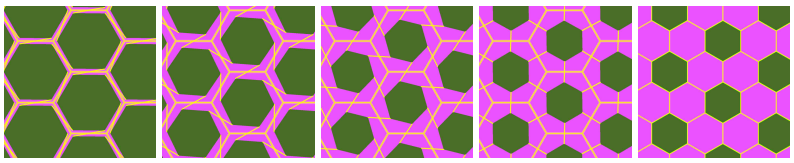


Figure 5: *How background grid of yellow outlined hexagons depends on the floor of t until the grid changes abruptly in last image of sequence to the new smaller grid, rotated through* 30° *and scaled by* $\sqrt{3}$ *compared to the initial grid.*

depend on the floor of the iteration level. This can be explained by the method of producing this image. In Figure 4, the initial image area is divided into a grid of hexagons. In the first image, each hexagon has a yellow boundary. This is the boundary of the grid system that the image is divided into. This division remains constant as the green hexagon is rotated through the next three images. The third and fourth image only differ in that the grid system is changed, so in the fifth image, a new grid system is introduced, with the new boundaries shown by the yellow lines. Therefore there is a jump in the grid system. If these boundaries are not shown, for example, in the top image of Figure 6, then the change across the image is gradual. The top image shows the yellow link lines, which change continuously, but not the underlying grid system. In the bottom image shows the grid system, which changes abruptly. Some images in this paper have aspects which depend on the floor of the iteration level, and when the level varies across the image, there may be abrupt changes. The code could in most cases be rewritten to have a gradual change, if desired.

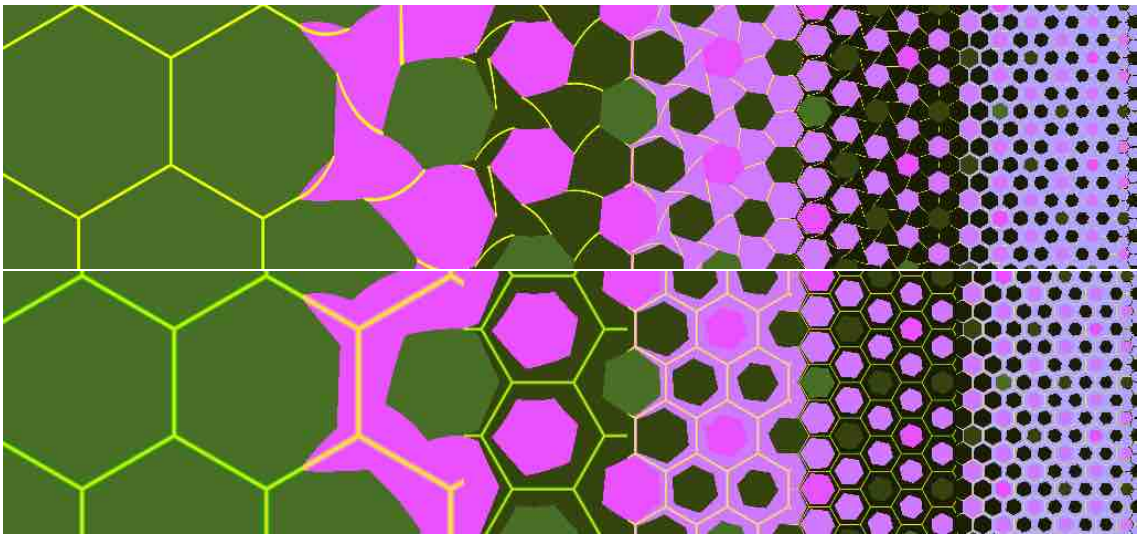


Figure 6: *How background grid jumps as iteration varies across image.*

1.2 Paths on Tiles

Now we add the Truchet tile pattern. Figure 12 is a more colourful version of Figure 6 in the main paper [7]. Operation 0 applied three times is shown in Figure 13. Figure 14 continues the sequence from Figure 6 [7]. Notice that in this figure, the red curve has a threefold rotational symmetry. This symmetry remains as further operations are applied, since the entire pattern in Figure 6 (c) [7] has a three-fold symmetry about the centre of the red curve, which is also the centre of a hexagon. Each operation itself, as seen in Figure 5 [7] and Figure 7, adds tiles which have a three fold symmetry about the centre of any original hexagon (the gray hexagons in the figures), so provided the original hexagons have a symmetry about the centre of a given hexagon, so do the original hexagons after the operation, since they are all rotated in the same way. Hence the whole pattern will retain a symmetry about the centre of a hexagon.

For clarity, Figure 7 shows the new tiles (the orange and blue ones) for operation 1, as a complement to Figure 5 in [7]. Figure 8 shows the operation as discrete replacement rules, including the Truchet design. In Figure 8 the background tile colour is chosen to correspond to the the orientation of the tiles; different background colour rules are used in most other examples. The rules for tile colour and arc colour are generally treated separately, so we can "mix and match". This figure focuses on arc/path transformation.

Figure 8 gives the result of the operations assuming all arcs are a single colour. In practice, I want to have differnt connected paths of distinct colours, as in the examples, such as in Figure 12. In this case, there are various possible replacement rules. Figure 9 shows a natural choice of replacement rule. As long as there is no change in colour along a path in the original tiling, there will be no change in colour along a path after the application of this operation. However, in the case where we colour each arc a different colour, with the simple rule of Figure 9 we end up with a situation as in Figure 9, with an unpleasant dividing line in arc colour across the new tiles, as in Figure 10 (c). The solution I choose in the initial WebGl program was to choose all added path segments to have a constant colour, chosen in a consistant way, as for example, in Figure 11 (a). This means there is no colour change within the new tiles, i.e., on the gray regions in this figure. In retrospect, it would have been better to choose a replacement rule as in Figure 11 (b). This is an option for a future rewrite of the program. Another possible solution to the iteration choice for paths with colour change could be to only allow colour change within tiles, not at boundaries.

As long as the entry and exit points on the tiles remain the same, ie., the replacement rule respects the path end points on tile boundaries, and replaces tiles with tiles with the same kind of path, the exact path

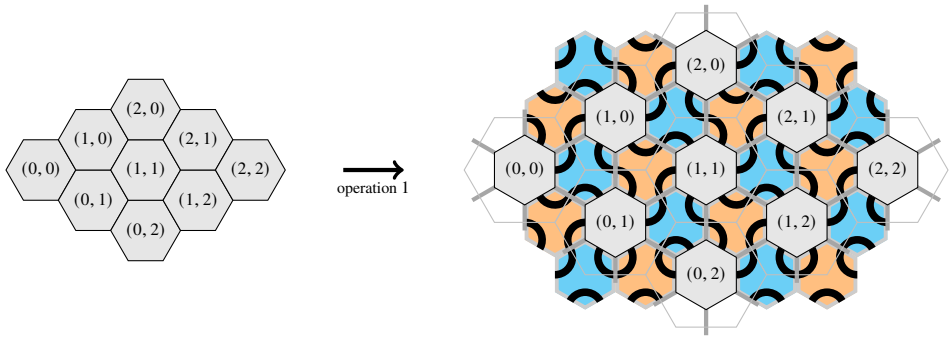


Figure 7: *Diagram showing the tiles added after applying the hinging operation* 1.

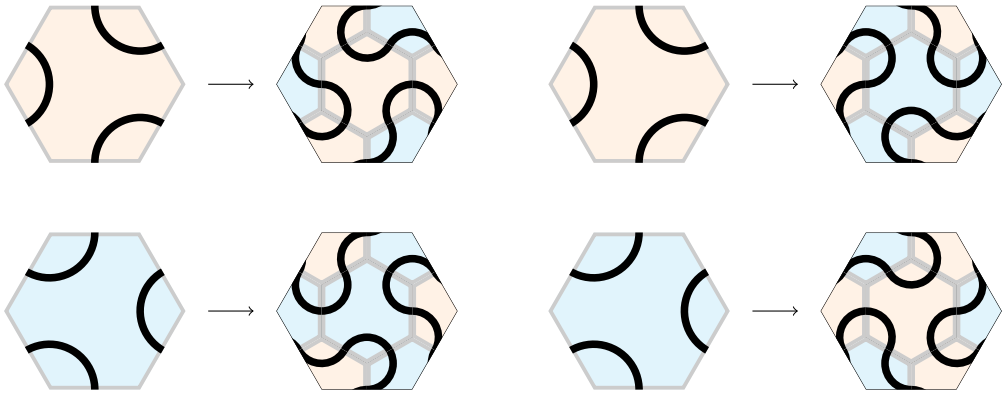
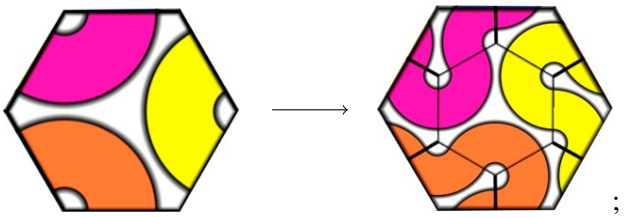


Figure 8: *Discrete replacement rules for tiles with Truchet design.*



;

Figure 9: *Replacement rule for coloured arcs, operation* 0*, tile orientation* 1*. (Tile orientation as in Figure 2, [7].)*

(a) initial state **(b)** after tile replacement **(c)** after removal of old tile divisions

Figure 10: *Replacement rule in Figure 9 applied to several adjacent tiles.*

(a) Constant colour on added paths on new tiles. **(b)** Balanced colours, extending both sides of old tiles.

Figure 11: *Alternative iteration replacement systems for coloured paths, used in preference to Figure 10 (c). The left version was used in the current program. Note that neither of these corresponds to a single tile replacement rule as in Figure 9; arc colours depend on more than one parent tile's arcs. New tiles have gray background for clarity.*

Figure 12: *Two applications of operation* 1*, with intermediate steps shown.*

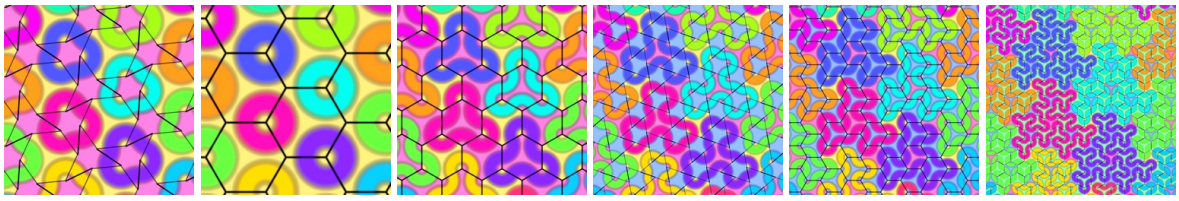


Figure 13: *Increasing iteration level, with values* $0, 0.3, 1, 1.5, 2, 3$ *respectively.*

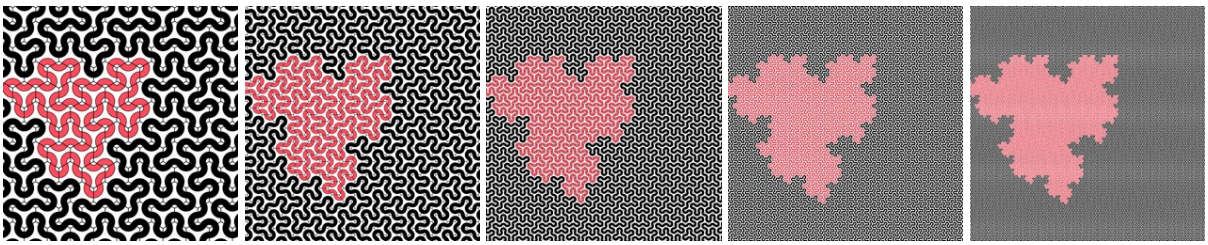


Figure 14: *A few more steps in the sequence started in Figure in [7]. The operation is an alternating sequence of* 0 *and* 1*, and the iteration stages are* $t = 3, 4, 5, 6, 7$ *respectively.*

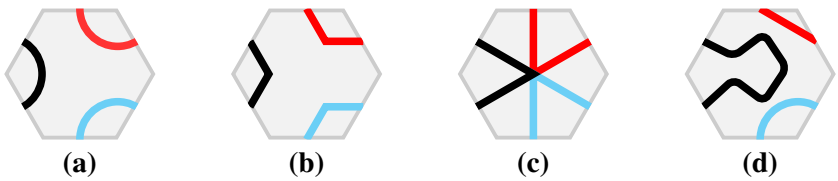


Figure 15: *Examples of different path styles on hexagon tiles.*

on the tile doesn't make a huge amount of difference after the iteration procedure is applied many times. Figure 15 illustrates four possible tile designs. Figure 15 (a), is our main choice. Figure 15 (b) is used for obtaining tiles as in Figure 19. Figure 15 (c) is best for proving that these curves are terdragon curves in the limit, since in this case the replacement rule will be exactly the same as the original replacement rule [1, p. 197]. This also has a nice feature of crossing tile boundaries at right angles, which (a) also has, but (b) does not have. Note that for the system given by (c), the limit curve passes through every point in the region within the boundary. for other versions, since we *never* pass through the centre of the hexagon, the resulting curves are not space filling, since although they pass arbirarily close to points at centres of hexagons, they never pass through these points, so the resulting sets have a lot of small holes. Since we always thicken our paths, as an artistic endevour, this is not an issue. Artistically, or perhaps pedagogically, the problem with (c) is that all paths meet at the central point of the hexagon, so you can't see what's going on. So, (c) may be best mathematically, but worst artistically, which is an example of potential tension between mathematical beauty and visual esthetics; both of which may be considered subjective, so it's not clear there is a solution to unite these. Figure 15 (d) is just to point out any curves would work, we just need to fix the end points on the tile boundaries.

In Figure 16, the colour, style, original paths, are the same, but the sequence of operations varies, as shown in the captions. In Figure 17, we see how increasing the line thickness of the curves leads to tilings.

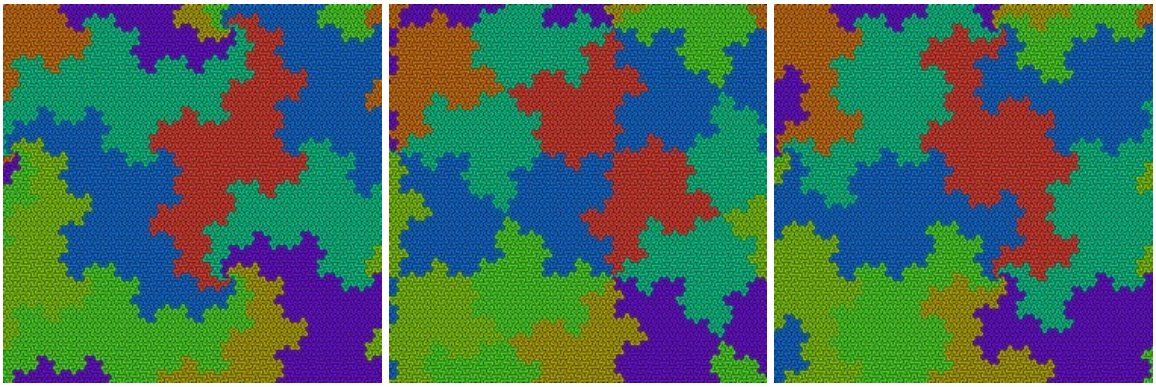


Figure 16: *Operation sequences 0000000, 1010101, 1100000 respectively.*

We can also use straight line edges for tiles. Figure 18 shows a comparison between straight and curved paths.

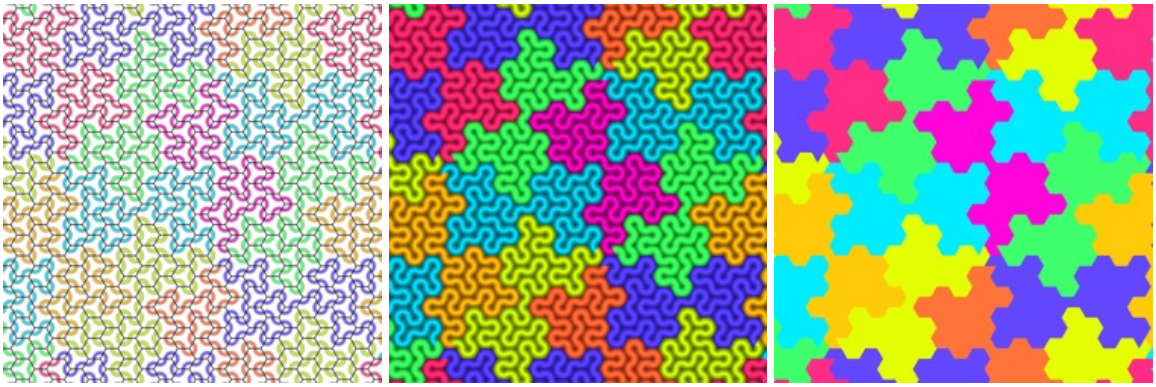


Figure 17: *Varying line width.*

Some examples with different operation sequences are shown in Figure 21. All of these are after four

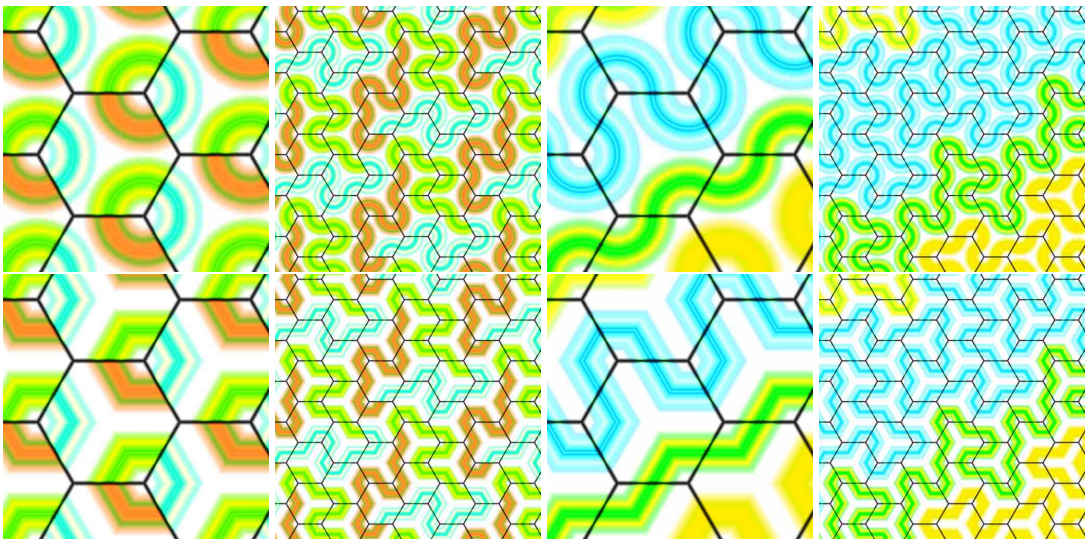


Figure 18: *Comparison of straight and curved line segments. Curved above and corresponding straight paths below. The iteration levels are either* $t = 0$ *or* 2*. Two different initial orientations are used.*

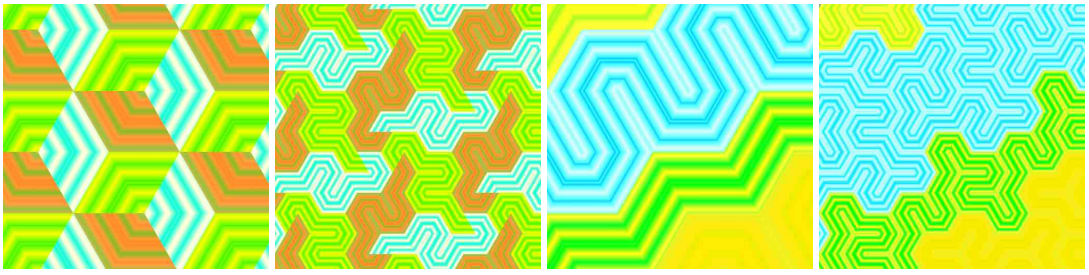


Figure 19: *Thickening of paths in second row of Figure 18 in order to obtain tiles with no background showing, and hexagon divisions removed.*

Figure 20: *Variable level of iteration across image.*

operations; this is the complete set, corresponding to all binary sequences of length 4.

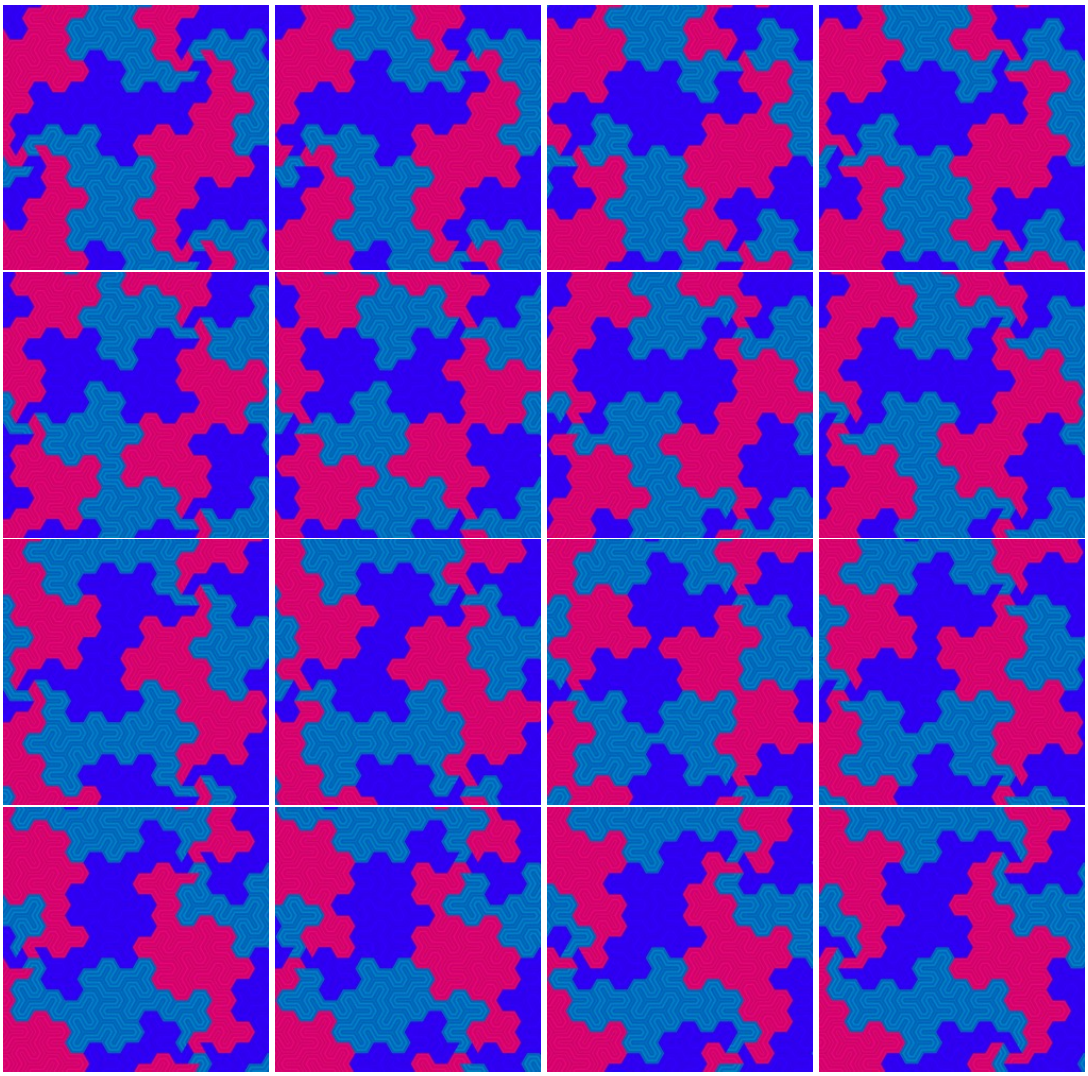


Figure 21: *All tiles obtained by 4 iterations, from 0000 to 1111.*

If we vary the level of iteration across the image, we obtain images such as in Figure 20. Note that in these images, the operation sequences are the same over the image (but different for different images). It is possible to smoothly transition from one operation sequence to another, but we would have to transition back to the initial path of the sequence for which they are the same. The fifth and sixth images differ in that fifth has variable iteration level, but the sixth has constant level of iteration across the image.

In [7], a discussion of why this construction gives rise to the terdragon curve is given. The terdragon, due to Davis and Knuth, and discussed in [3], corresponds to the application of 0 arbitratily often.

1.3 Koch Snowflake Background

As described in the main paper [7], we can obtain a Koch snowflake background. Some examples are shown in Figures jpgs/27, 24, and 25. In [7, Figure 11] an algorithm is given to describe the colour of a point in the image. The algorithm needs a formula to give a colour for each integer, up to how ever many iterations of operation 0 or 1 are applied. For the images in the program, for this particular background style (as opposed

to the constant tile colour background) I use a formula

$$\text{colour}(c) = (170, 170, 170) + 170\sqrt{1.5/6}\cos(\gamma\pi)(1, 1, -2) + 170\sqrt{1.5/2}\sin(\gamma\pi)(1, -1, 0), \qquad (1)$$

where the colour is given by three component vector describing red, green, blue components as real values between 0 and 255, $\gamma = \alpha c + \beta$, and α and β are parameters which can be controlled by the user of the program. Any other formula could be used, this is just to give a concrete example. For example, in Figure 22 a modulo 2 colour scheme is used, where the colour is yellow for c even, and blue for c odd. In Figure 23, $\alpha = 1.1$ and $\beta = 1$.

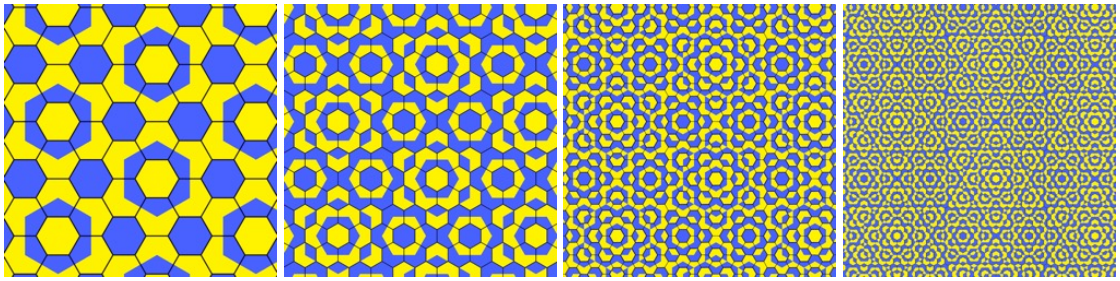


Figure 22: *Koch snowflake modulo* 2 *colouring scheme for* $t = 2, 3, 4, 5$.

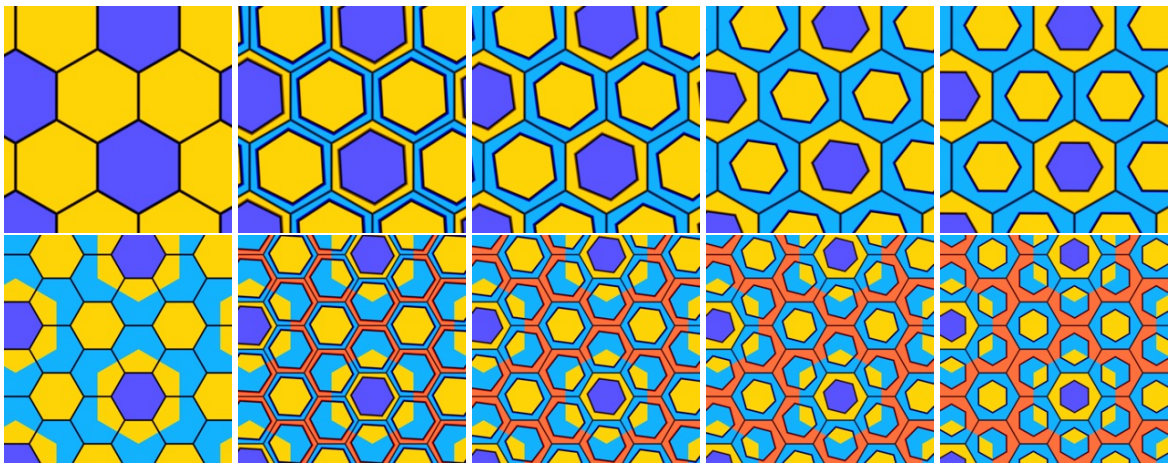


Figure 23: *Development of Koch snowflake pattern, with* $t = 1, 1.25, 1.4, 1.8, 1.99, 2, 2.25, 2.4, 2.6, 2.99$.

2 Hinged Truchet Triangles, Version 1

The first version of the triangle hinging is illustrated without the Truchet design in [7, Figure 13] with the $t = 1$ case shown just before adding triangle divisions in the hexagon areas in [7, Figure 13 (h)], and just after in [7, Figure 14 (e)]. In Figure 28 the hinging with the Truchet design is shown.

Images obtained from the first triangle hinged tiling variant are shown below. The concepts illustrated are similar to those for the hexagon case, and the method is described in [7]. Figure 29 shows the background tiles only. Note that for these tilings, when the arcs are added, each tile has one arc, between two of the edges, and one "dot", which is an end point of a path, on one edge, as seen in the examples in Figure 32. This means that at each iteration stage, more components (connected paths, disconnected from previous paths) are added to the image, which grow from dots to paths, all with the same fractal structure in the limit.

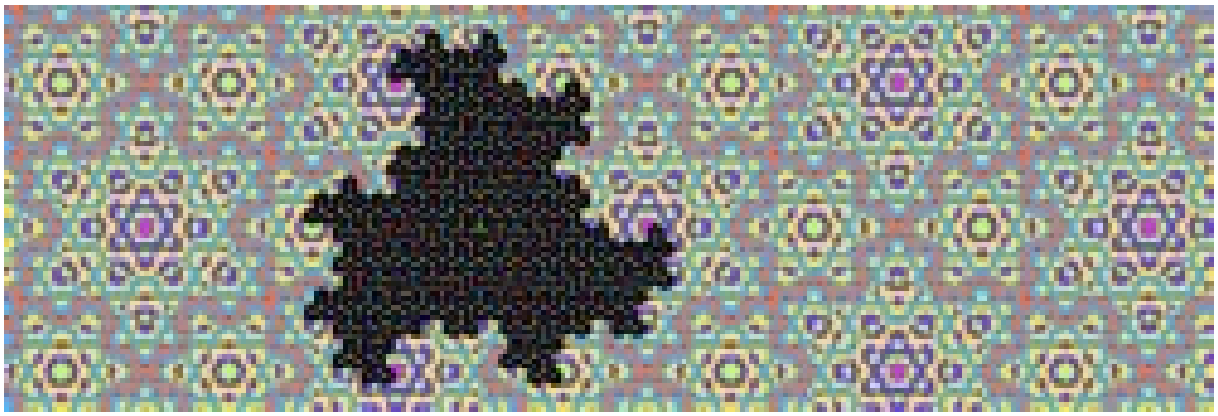


Figure 24: *Koch snowflake tessellation with terdragon curve union.*

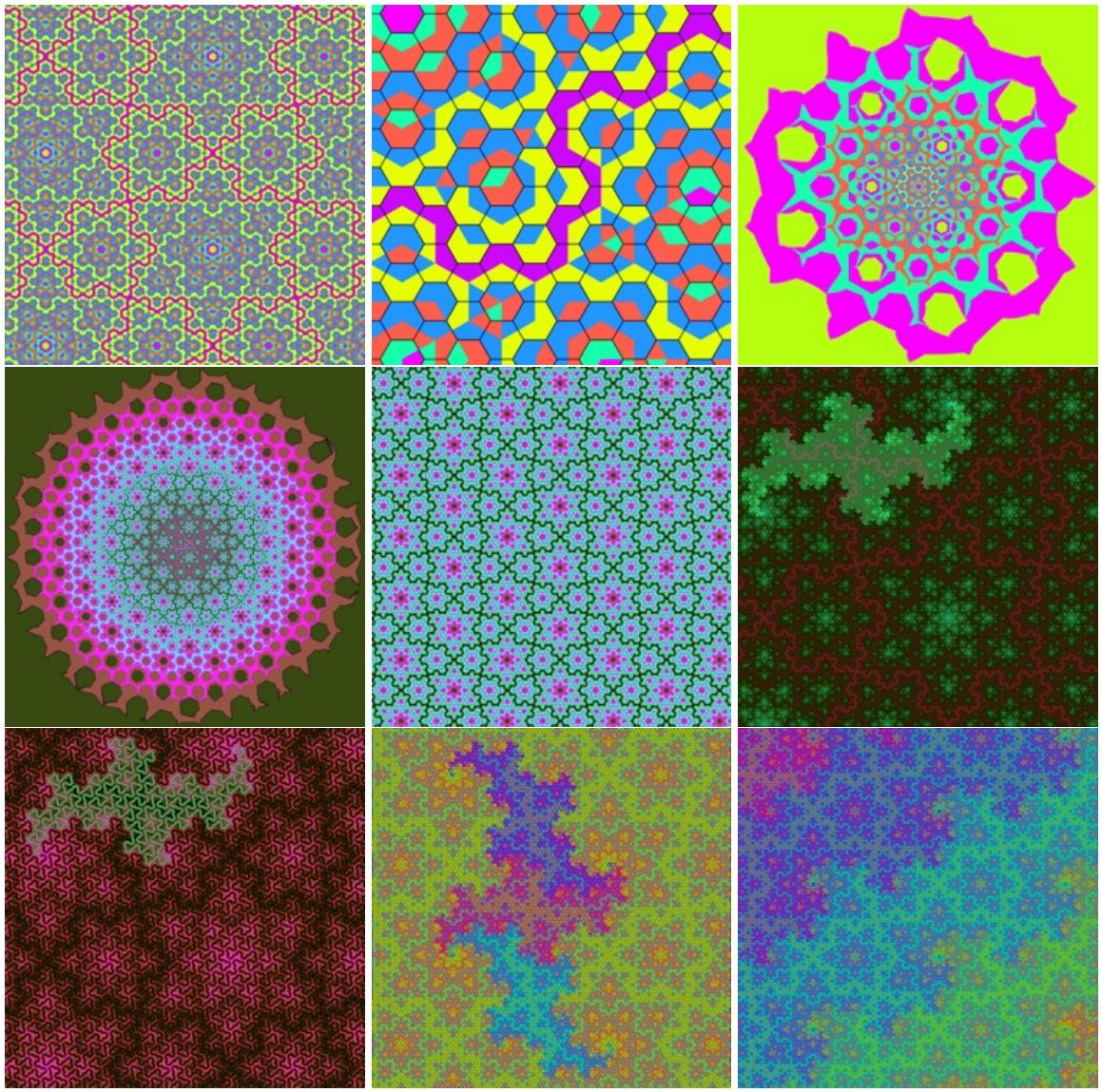


Figure 25: *Koch snowflake tessellation.*

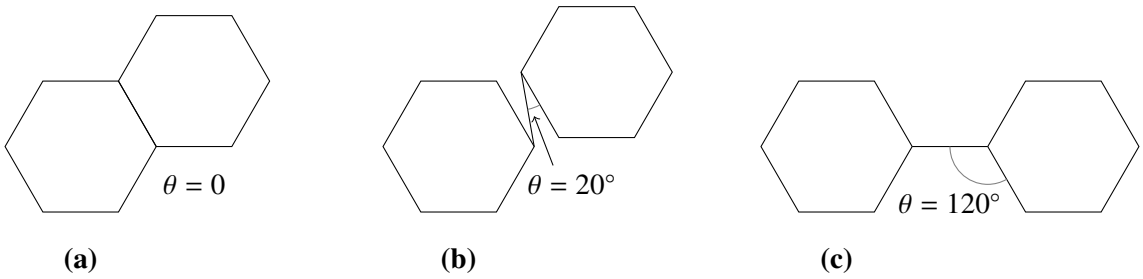


Figure 26: *Diagram of two tiles within hinged tiling with links. No scaling while hinging. In our implementation,* $\theta = 120\{t\}$*, where* $\{t\}$ *is the fractional part of* t*.*

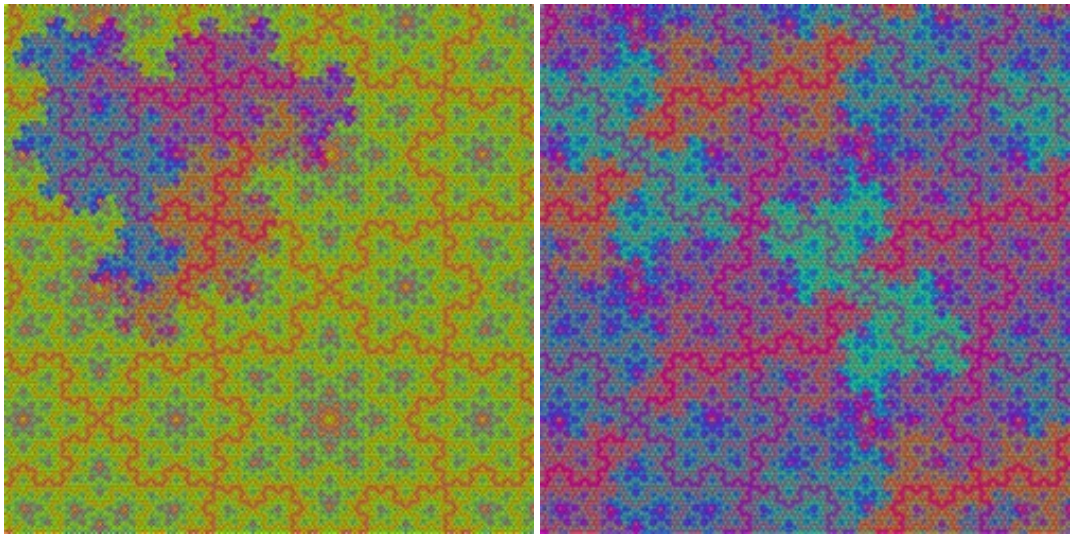


Figure 27: *Koch snowflake tessellation.*

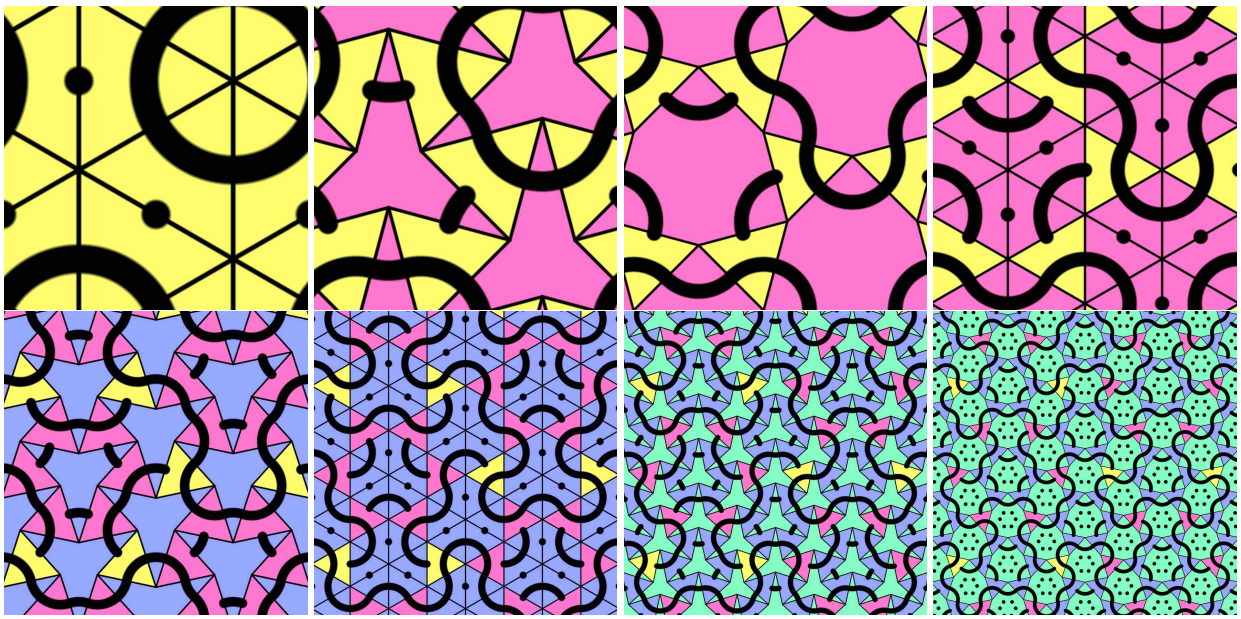


Figure 28: *Hinged triangle tiling version* 1*, with* $t = 0, 0.25, 0.75, 1, 1.3, 2, 2.2, 2.7$*, respectively.*

In Figure 31, the tile orientation corresponds to the tile background colour. From the fourth to fifth images in Figure 30, the only difference is change of the underlying grid system; the curves are virtually the same. The grid depends on floor of level, so different grid for level 0.9999 and 1, but the curves become arbitrarily close.

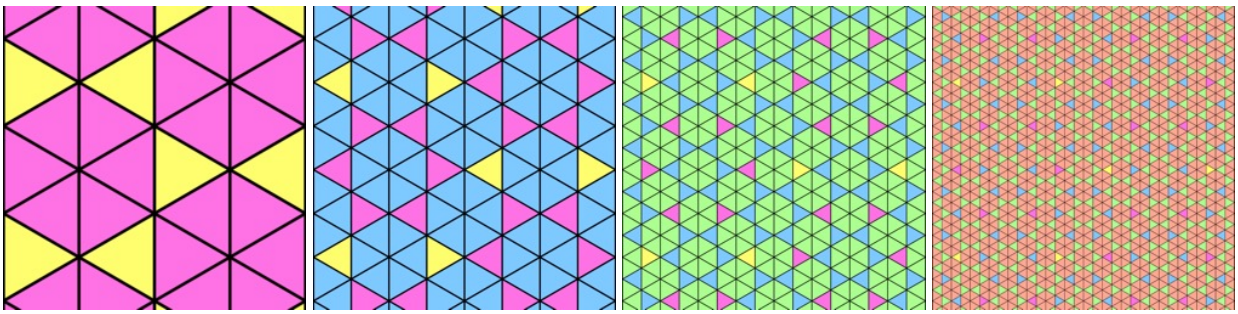


Figure 29: *Background tiles, iteration level* $1, 2, 3, 4$.

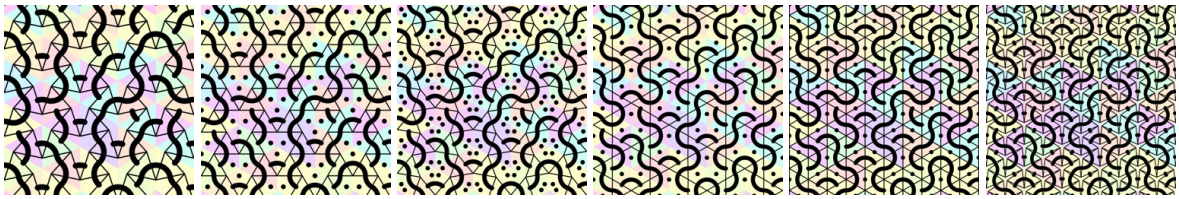


Figure 30: *Example with iteration stage* $0.3, 0.5, 0.66, 0.99, 1, 1.1$.

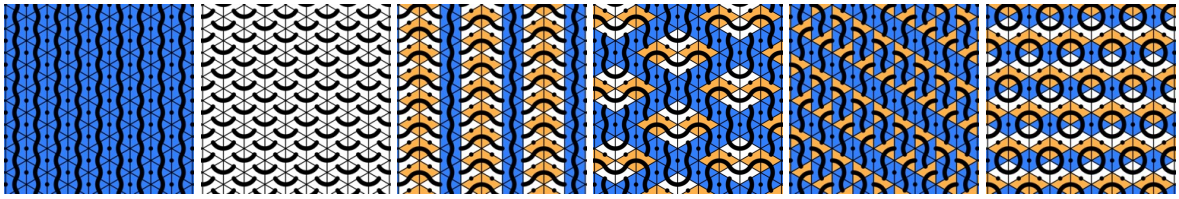


Figure 31: *Examples with background tile colour corresponding to orientation.*

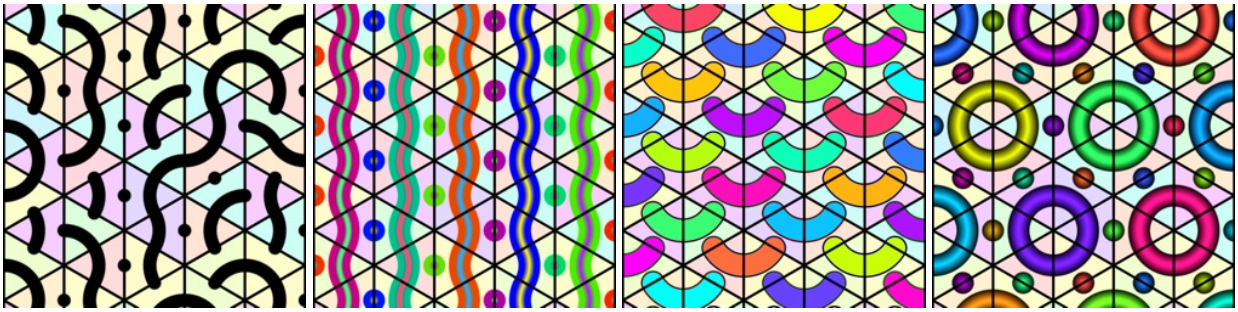


Figure 32: *Initial tiles example path styles.*

Figure 34 shows a close up of a figure similar to [7, Figure 16 (d)], showing iteration stage $t = 4, 4.5$ and 5 for comparison. In Figure 35 all the paths are shown. In Figure 34, any path not present in the initial figure is not shown. Otherwise, Figure 35 and Figure 34 are the same. The paths (or dots) that have been added at stages $1, 2, 3, 4$, and 5 are lilac, light blue, white, green, and red repsectively.

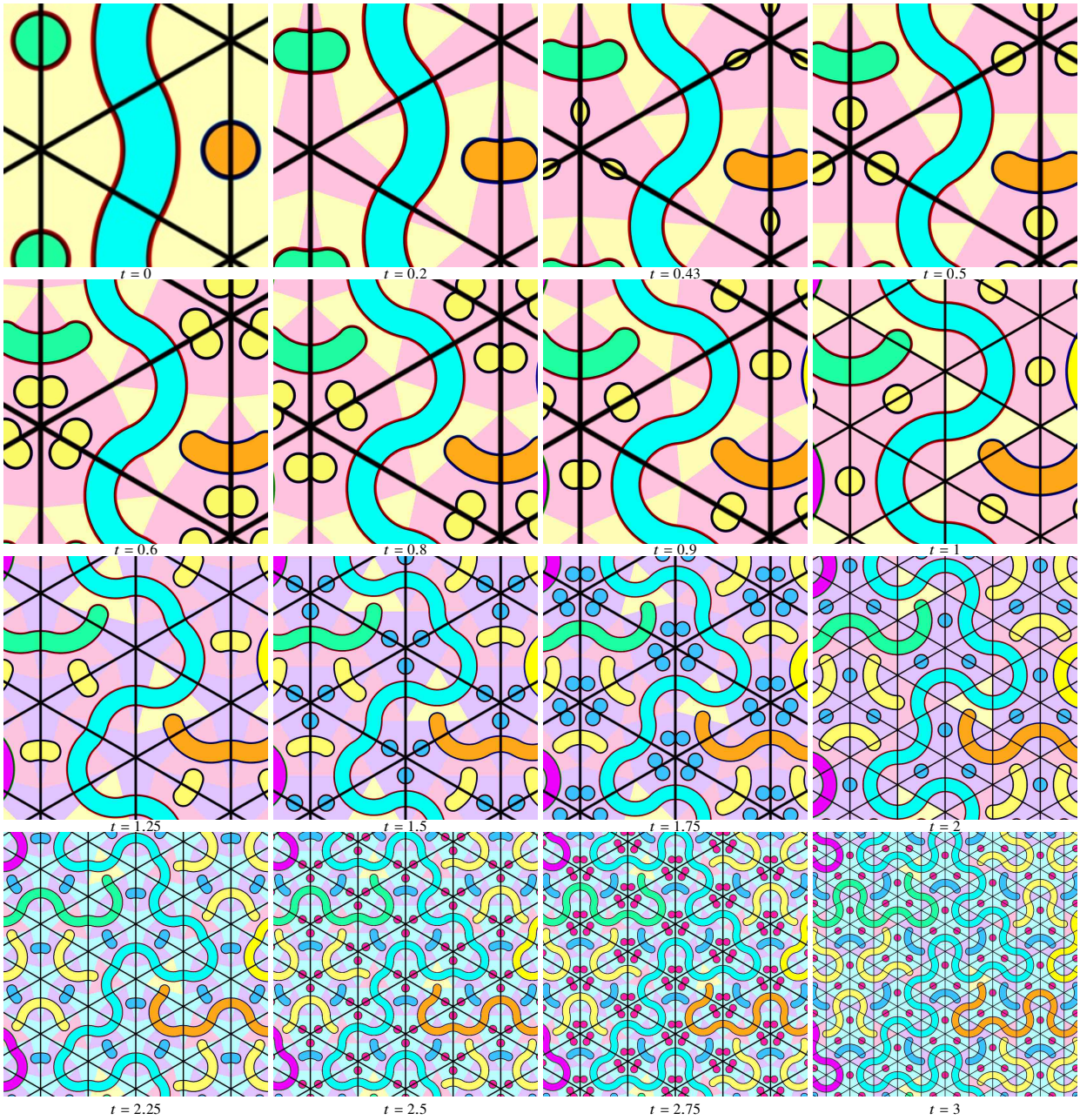


Figure 33: *Application of operation* 1*, followed by operation* 0*, followed by operation* 0*. Notice that three yellow dots appear on the left side at* $t = 0.43$*. These separate out into* 6 *dots by* $t = 0.8$*, and then come togther again at* $t = 1$ *to form three dots again, in different places to the three at* $t = 0.5$*. From* $t = 1.25$ *to* $t = 2$ *these spread out into little arcs, and new, blue dots appear. For this implementation, the initial paths and dots have different colours, but subsequently, I have chosen to have the same colour, depending on n, for all new paths (initially dots) introduced at stage* $t = n$*.*

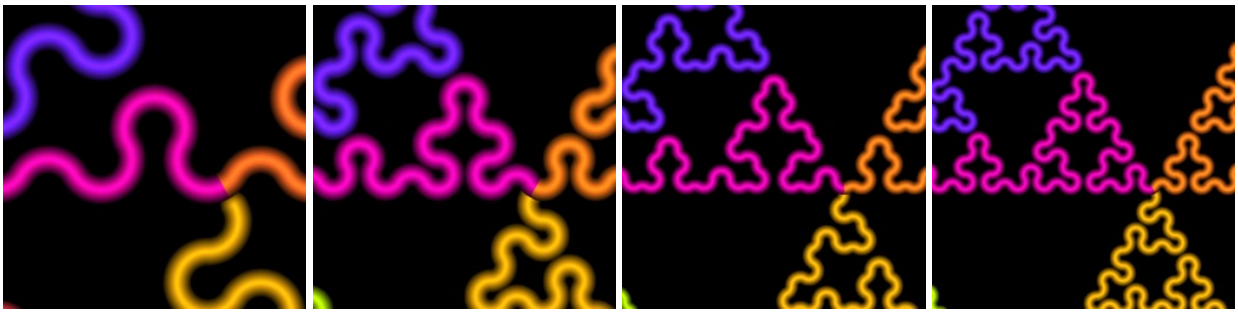
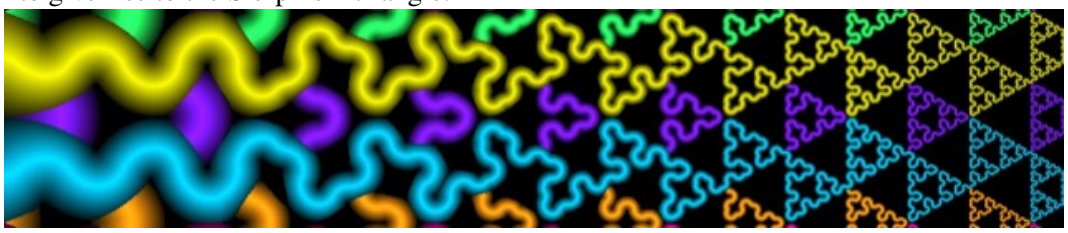


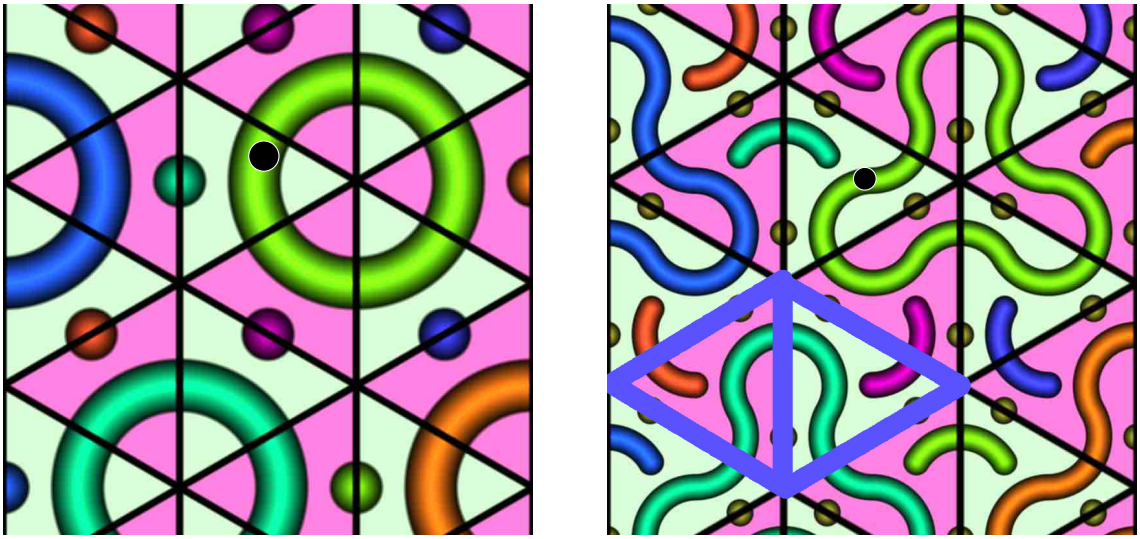
Figure 34: *Hinged triangle version 1, operation* 0 *applied* $3, 4, 4.5$*, and* 5 *times.*

Figure 35: *Hinged triangle version 1, operation* 0 *applied* $3, 4, 4.5$*, and* 5 *times.*

Now we explain why the Sierpinski triangle is produced by this method. As in [7, Figure 15 (c)], after operation 0, tiles have neighbours as in [7, Figure 14, right]. So all original (gray) even tiles must be entered traveling from the left, and departed traveling from the right, and the converse for odd tiles. Also that all even tiles (triangles pointing right) become odd tiles, and odd tiles become even tiles. Therefore, we can specify the corresponding L-system as follows. An initial path can be given as $XeXoXeXo\cdots$, where X is L (left) or R (right), and e and o mean passing into an even tile, or passing into an odd tile respectively, so, moving a unit length. (Note that $X \neq X$ for X in different positions; X is just a place holder here, not a fixed symbol.) So for example, following the green circle in Figure 36 (a), in a clockwise manner, and starting from the black dot on the left side, can be described by the sequence o**R**e**R**o**R**e**R**o**R**. After applying operation 0, the transformed version, in (b), also starting from the black dot, is given by eLoLe**R**oReRo**R**eLoLe**R**oReRo**R**eLoLe**R**oReRo**R**. The bold letters are to indicate where the old tiles end up in the sequence. Since the paths on the original tiles still turn in the same direction after rotation, the L-system replacement rule includes $L \to L$, $R \to R$. Since tiles are rotated to point in the opposite direction, the parity of tiles is reversed. Thus crossing from odd to even, e, is replaced by crossing from even to odd, o, followed by two right turns, ie. $e \to oReRo$, since we enter the now odd original tile by turning right. Similarly $o \to eLoLe$. We can compare this to the description in [11] to see that we have an L-system which is known to give rise to the Sierpinski triangle.

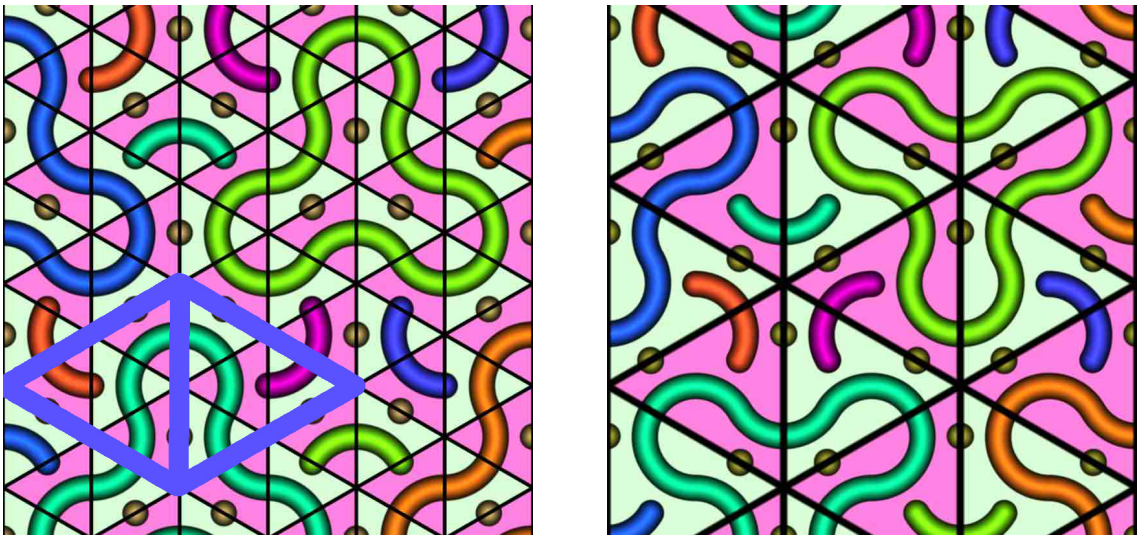


;



(a) **(b)**

Figure 36: *Application of operation* 0*, for examination of how the turning instructions are transformed. Tiles are coloured according to parity. The right figure is just before reassignment of parity to new subtiles of the original tiles.*



(a) **(b)**

Figure 37: *The left figure illustrates how the parity of subtiles depends on the parity of the original tile. This is Figure 36 (b) after tile division at the end of application of operation* 0 *to Figure 36 (a). In Figure 36 (b) two tiles are outlined in thick blue lines for easy visual location and comparison with the tiles after division, with the same outlined area in (a) in this figure. The right figure gives a comparison of Figure 36 (b) (operation* 0 *from Figure 36 left) with the result of operation* 1*.*

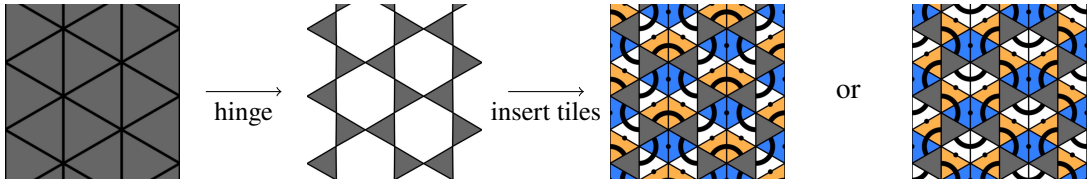
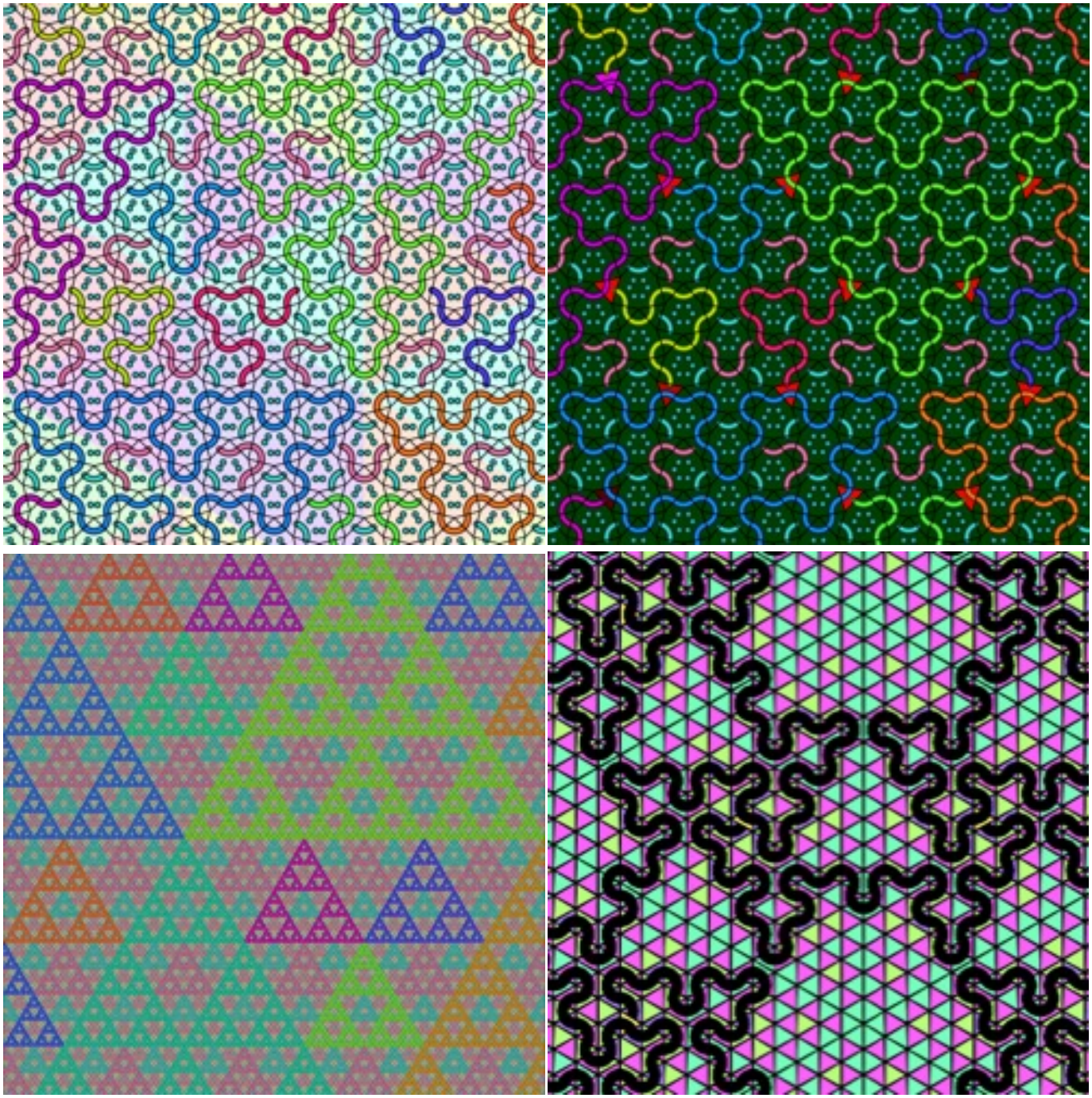
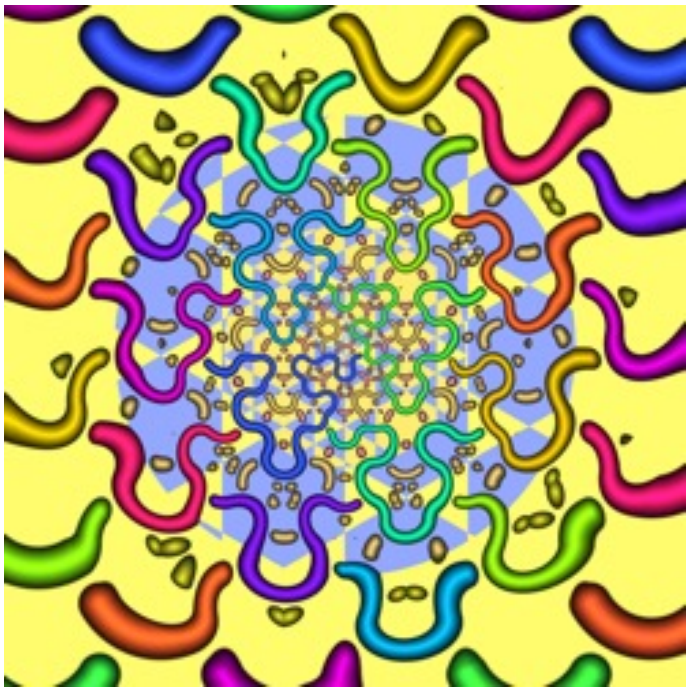
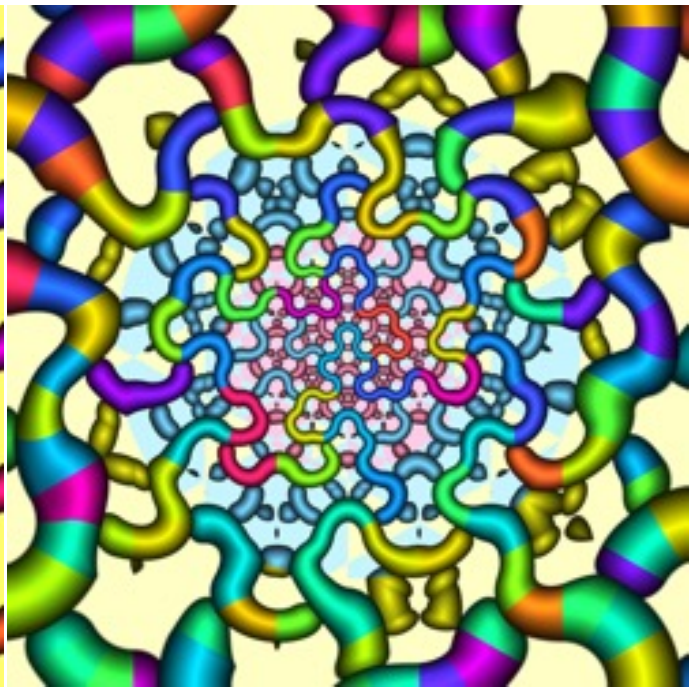
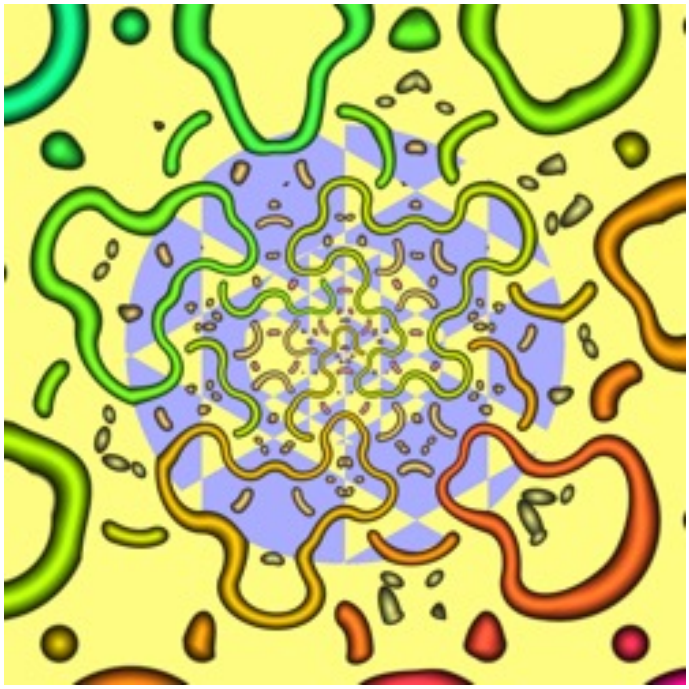
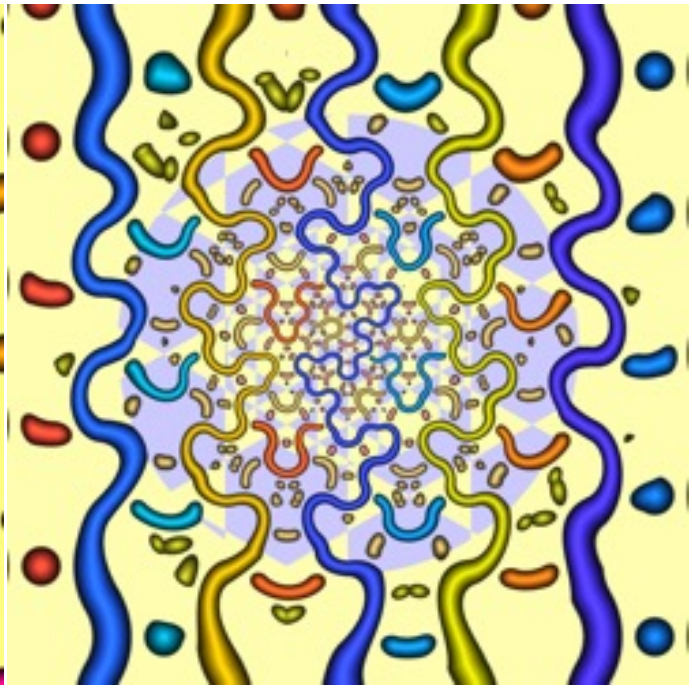
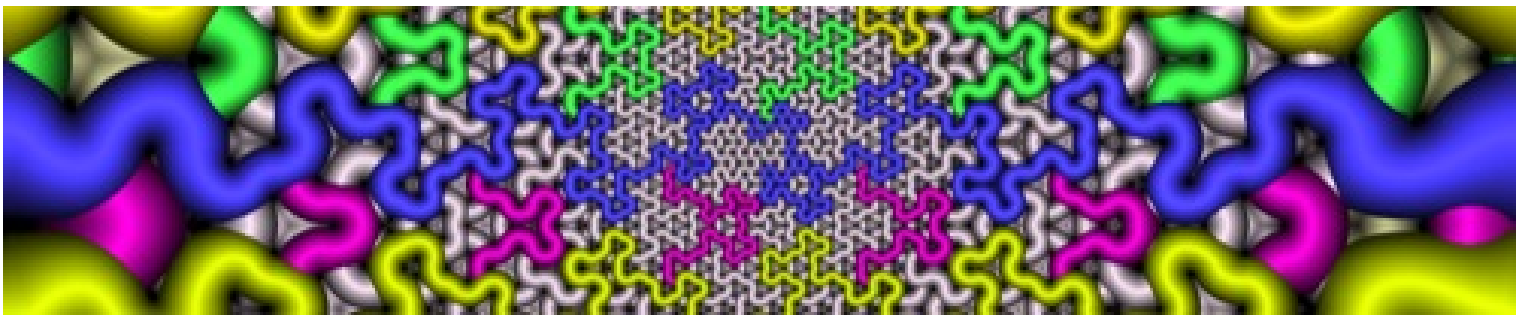
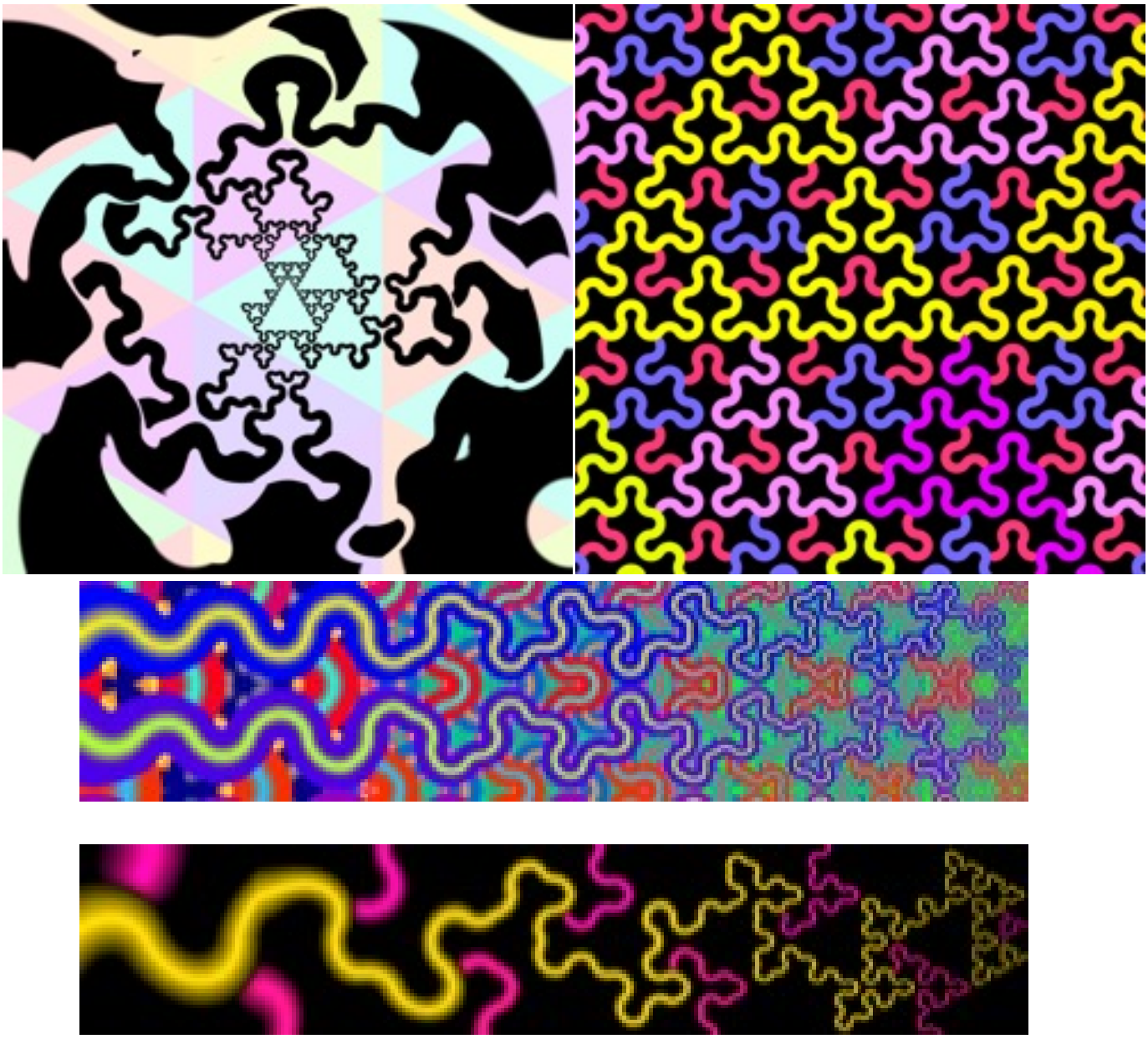


Figure 38: *Insertion rules, for operations* 0 *or* 1 *respectively.*





3 Hinged Truchet Triangles, Version 2

This section is an alternative hinged Truchet triangle fractal procedure. Originally this was planned to be in the main paper, but there was not sufficient space.

In this version of the iteration step, instead of rotating tiles through 60 degrees, we stop the rotation after 30 degrees, as in Figure 39. Again, there are two operations, 0 and 1, corresponding to the two directions of rotation. The gap is now a triangle shape, which we divide into 4 triangles. Now we apply the Truchet design and hinge operation. There is a unique way to insert three of the the four new tiles in order to preserve paths. The path on the middle inserted tile is not connected to the original paths, and can be inserted in any way. This will affect the orientation, but not the shape of the newly introduced paths. I have chosen to insert all of these tiles in orientation 0 in [8]. The first two applications of operation 1 are shown in Figure 40 (b) and (c), starting from (a). Nine applications of operations 0 (to a different starting image) gives (d). The L-system

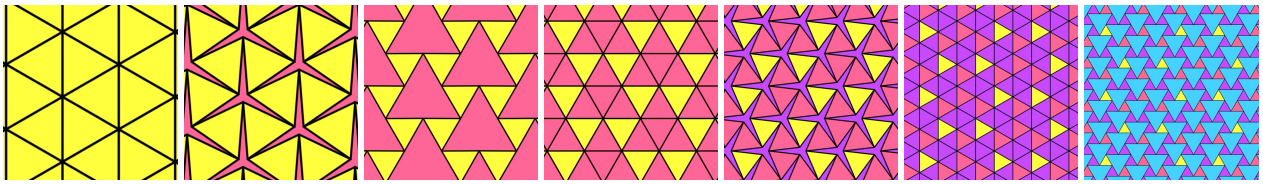


Figure 39: *Hinged triangle operation. Replacement rule applied after hinged rotation through* 30 *degrees.*

is very similar to the previous triangle hinged tiling. An initial path still has the form $XeXoXeXo\ldots$. But now instead of two, we only have one new tile between each old pair of previously adjacent edges. The L-system now is given by $R \to R$, $L \to L$, $o \to eLo$, $e \to oRe$. We recognise the fractal produced by the

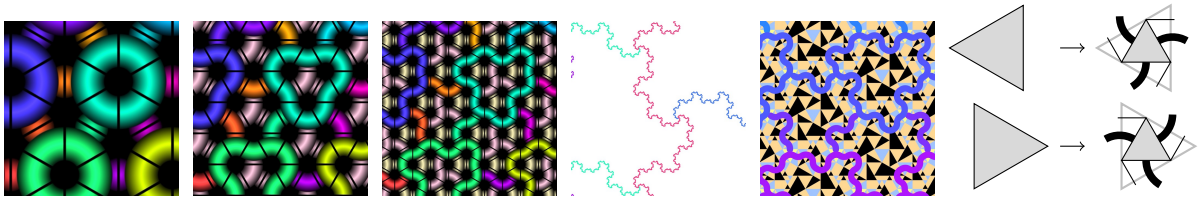
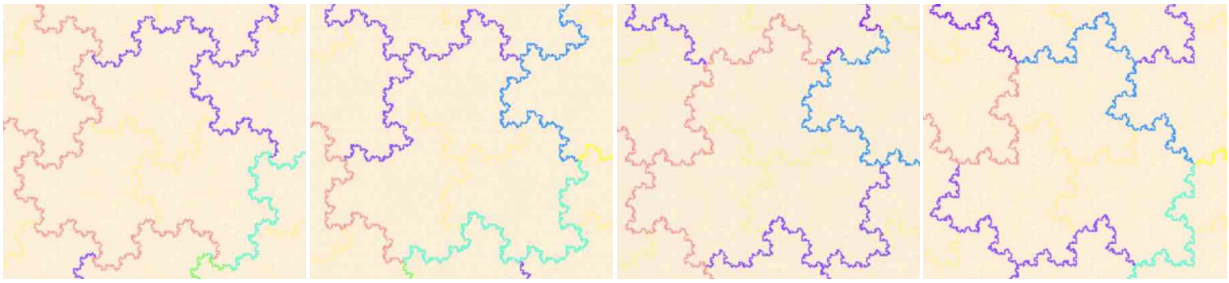


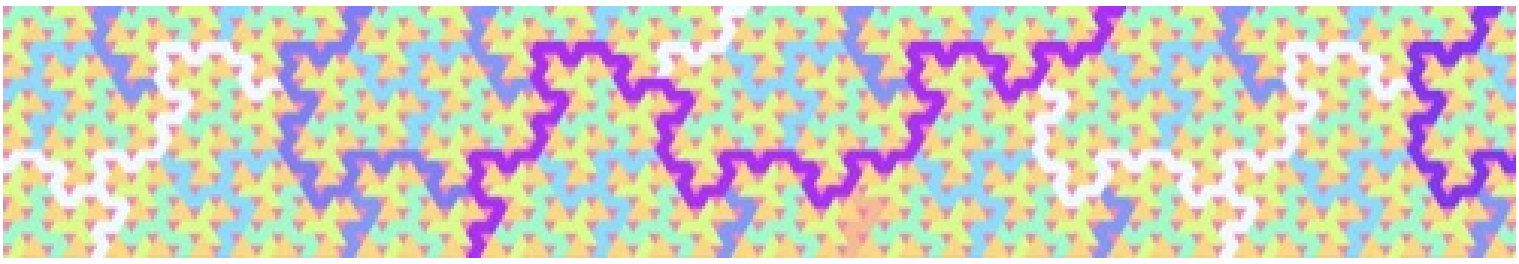
Figure 40: *Hinged triangle Truchet tiles producing terdragon boundary.*

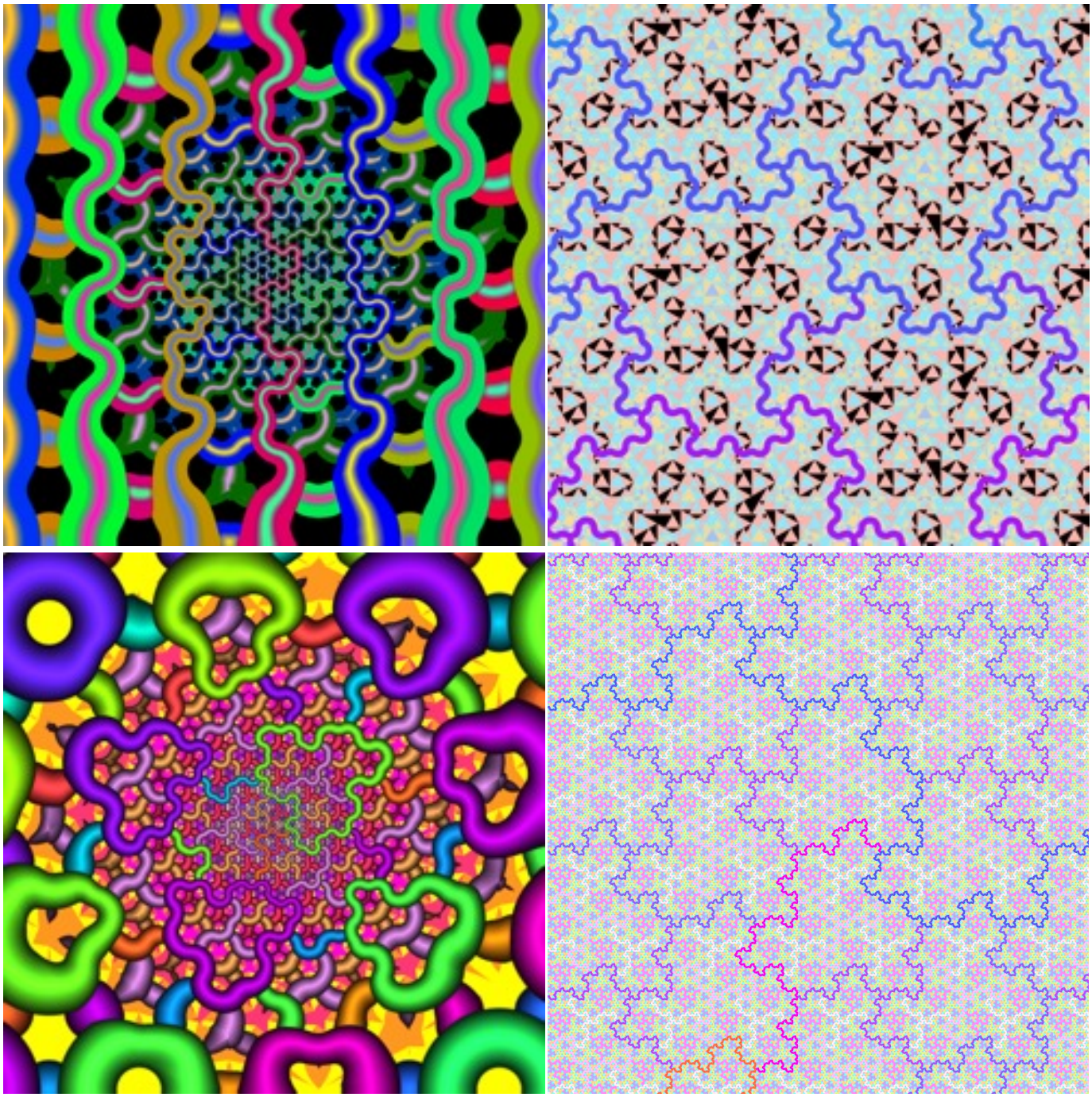
initial curves to be the boundary of the terdragon curve. This fractal is known as the "fudgeflake", when three copies of the limit path of a single arc are take. Further details of the Fudgeflake are given in [1, pp. 23 and 39], and [4]. An example of the same background colouring algorithm of [7, Figure 11] applied to this tiling is shown in Figure 40 (e).

4 Triangle Tiling 2, Images

Images obtained from the second triangle hinged tiling variant are shown below. The first row of four images correspond to the sequences of operations $1111\ldots$, $100100100\ldots$, $00110011\ldots$ and $101010\ldots$ respectively.







5 The Boundary of the Terdragon Curve

In this section we discuss the relationship between the hexagon hinged tiling, which gives rise to the terdragon curve and relatives, and the triangle 2 hinged tiling, which gives rise to the boundary of the terdragon curves, in particular, including the fudgeflake curve. See [1, pp. 23 and 39], and [4].

The hexagon hinged tiling and the second triangle hinged tiling are given as in Figure 41. These are taken from [7], Figures jpgs/5 and 14, where there are further details of these replacement rules. We only show one of the two possible operations 0 or 1, which correspond to the rotation direction. Since the result

for the other operation differs by a symmetry, we only need to cover one case. We also only show the odd triangle (pointing left); the even case is the mirror image. In [7] Figure 5, it is shown how the hexagon tiles

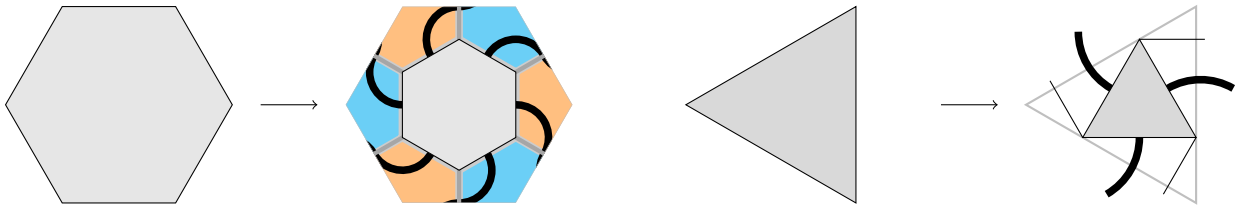


Figure 41: *Diagram showing the tiles added after applying the hinging operation.*

fit together. Since there is not enough space in [7], how the triangles fit together is shown in Figure 42 for clarity.

In Figure 42, the even tiles are yellow, odd tiles pink; these are the initial tiles, and can be any orientation when the Truchet design is added. From the left side to right, each tile is scaled by a factor of $1/\sqrt{3}$. The yellow tiles rotate 30 degrees clockwise, and the pink tiles 30 degrees counterclockwise. They all end up in the same orientation. There are four tiles to be added in the gaps. The middle tile (middle as in middle of the gap that opens up when hinged, not the new smaller tile in the middle of the original tile) can be added in any orientation; these tiles are drawn in gray, with no Truchet design. The the other, in white, have the given orientations (curve choices), since these orientations are required to preserve connected paths, whatever those paths are. Note that the white triangles are split over two original tiles, and the gray over 6 original tiles; in the algorithm, after getting to this step, we redivide the plane, to glue these pieces into whole tiles, i.e., after a change of grid, we won't actually show the thick dividing lines in the second picture.

In Figure 43, the tiles of Figure 41 are superimposed, to show their relationship. We choose a compatible (i.e., with lines on triangle and hexagon tiles non-intersecting when superimposed) orientation for the original tiles, shown by the pink lines on the triangle, and blue and green on the hexagon. Just one case is given, since the other cases are equal up to rotation. We see that the paths on the hexagon and triangle do not intersect, either before or after the operation. I.e., non-intersection of paths is preserved by the operation. Moreover, if we consider two colours of paths in the hexagon tiling (e.g., green and blue), as in Figure 43, with a curve (pink/red) on the triangles adjacent to two colours before the operation, then it is still adjacent to these two after the operation, i.e., one colour on the left, one on the right of the curve. (Note that the yellow path will be ignored; only a single point of this touches the red/pink path, so this won't effect the fact that the red/pink path is a fractile boundary. If one side of the (red/pink) curve is adjacent to one curve (e.g., the green curve), and the other to the other (e.g., the blue curve), then this curve must lie along the boundary between where the tiles meet, i.e., this path does describe the boundary of a fractal terdragon curve, as it appeared to from

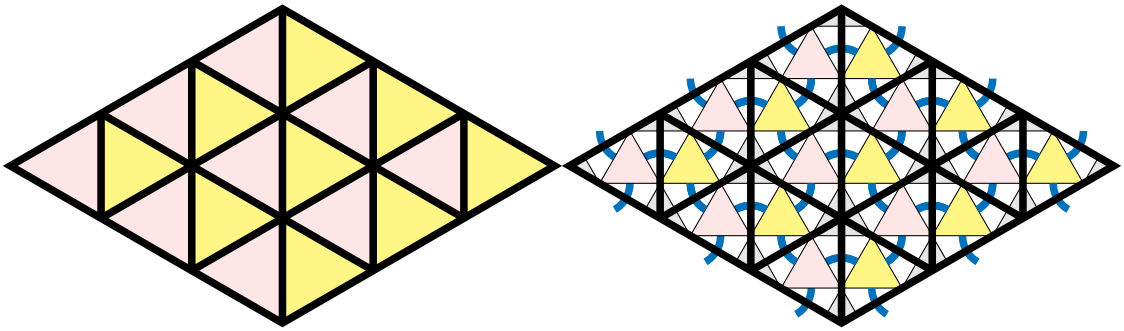


Figure 42

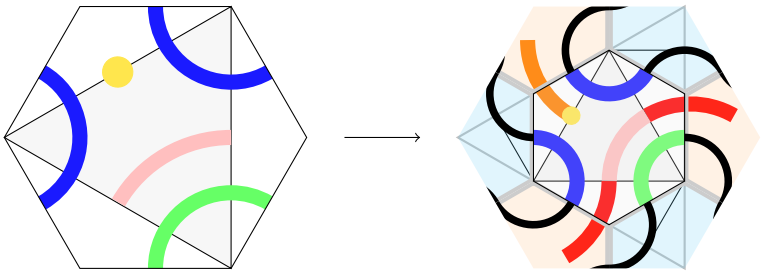


Figure 43: *Hexagon and Triangle hinged tilings units superimposed, counterclockwise rotation.*

the computer program. Given the description of the L-system given in [7], we can state this result as follows.

Definition: Define an L-system as follows: Symbols are e, o, L, R, and strings have the form

$$X_1 e X_2 o X_3 e X_4 o \ldots$$

where X_i is L or R. The symbols L, R remain constant, and o and e are replaced by either

$$\mathcal{L}_0(e) = oReRo, \quad \mathcal{L}_0(o) = eLoLe$$

$$\mathcal{L}_1(e) = oLeLo, \quad \mathcal{L}_1(o) = eRoRe$$

A relative of the terdragon curve corresponding to a binary sequence $e_1 e_2 e_3 \ldots$, is the limit of the curves corresponding to the sequences $\mathcal{L}_{e_n} \mathcal{L}_{e_{n-1}} \mathcal{L}_{e_{n-1}} \cdots \mathcal{L}_{e_1}$ applied to any starting sequence of this L-system. The curve corresponding to this L-system is given by interpreting o and e as "move forwards by $1/\sqrt{3}^n$" and L and R mean turn through 30 degrees to the left or right respectively, making a 60 degree angle in the path.

Theorem: Define an L-system as follows. Symbols are the same as for the terdragon case. Operations $\mathcal{L}_0$ and $\mathcal{L}_1$ are defined as follows. The symbols L, R remain constant, and o and e are replaced by:

$$\mathcal{L}_0(e) = oLe, \quad \mathcal{L}_0(o) = eRo$$

$$\mathcal{L}_1(e) = oRe, \quad \mathcal{L}_1(o) = eLo$$

Interpret the symbols to define a curve by the same rules as for the terdragon L-system. Take a binary sequence $e_1 e_2 e_3 \ldots$, and apply the corresponding operations $\mathcal{L}_{e_i}$ to a starting sequence. Then the resulting curve has limit given by the boundary of the corresponding relative of the terdragon curve.

References

[1] G. Edgar. *Measure, Topology, and Fractal Geometry*. Undergraduate Texts in Mathematics. Springer, 2008.

[2] A. Flores. *Hinged Tilings*. https://www.public.asu.edu/~aaafp/tiling/hingedtilingtext.html

[3] L. Riddle, Agnes Scott College, *Classic Iterated Function Systems, Terdragon*, https://larryriddle.agnesscott.org/ifs/heighway/terdragon.htm

[4] L. Riddle, Agnes Scott College, *Classic Iterated Function Systems, Fudgeflake*, https://larryriddle.agnesscott.org/ifs/heighway/fudgeflake.htm

[5] S. Tabachnikov. "Dragon Curves Revisited". *Math. Intelligencer*, vol. 36, no. 1, 2014, pp. 13–17.

[6] H. Verrill. "Space Filling Truchet Variations". *Bridges Conference Proceedings*, 2023, pp. 465–468.

[7] H. Verrill. "Fractals from Hinged Hexagon and Triangle Tilings" *Bridges Conference Proceeding* 2024.

[8] H. Verrill. *Hinged Truchet Tilings 2*, 2024. https://www.mathamaze.co.uk/Truchet2/.

[9] Wikipedia. *Dragon Curve*. 2024. https://en.wikipedia.org/wiki/Dragon_curve.

[10] Wikipedia. *Koch snowflake*. 2024. https://en.wikipedia.org/wiki/Koch_snowflake

[11] Wikipedia. *L-system*. 2024. https://en.wikipedia.org/wiki/L-system.