Addendum to Space Filling Smith Truchet Variations

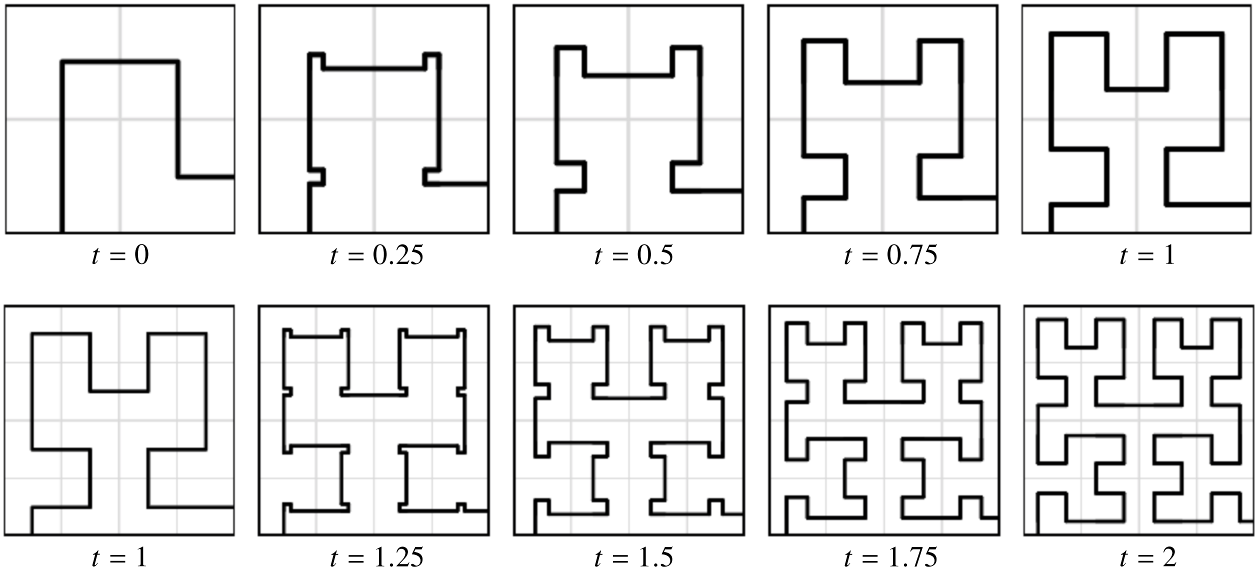
H. A. Verrill

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

This collection of images helps to clarify aspects of my 2023 Bridges paper, quoted here as [V]. (Available via https://archive.bridgesmathart.org/2023/)

Although the main idea of the paper is about Smith Truchet hinged fractal dragon tilings, it's also about continuous deformation of the process of producing a fractal space filling curve. In order to avoid introducing too many new ideas at one time, the discussion starts with a consideration of the Hilbert curve, since this may be more familiar to some readers. The concept of a continuous deformation interpolating between two integer steps of the iterative procedure is first discussed in this context. This continuous deformation method could be applied to any stepwise fractal creation process, you just have to make an appropriate interpolation.

Figure 3 (a) [V] shows a superimposed set of curves which are interpolations of the steps $t = 0$ to $t = 1$ in Figure 2 [V]. In Figure 1 these curves are drawn separately for additional clarity, and additional curves are shown, showing how after $t = 1$, we switch to the curve being divided into 4 quarters, with the deformation applied to each until we reach $t = 2$. This process continues for higher t, so the curve is constructed by diving the tiling according to $\lfloor t \rfloor$, and the stage of the deformation by $\{t\}$. Note that although the curve is deformed continuously, the division into tiles occurs discretely, with every tile divided into 4 smaller tiles at the point t becomes an integer, at which point the value of $\{t\}$, the fractional part of t, also makes a jump from 1 to 0. The deformation is devised so that the curve at time 1 is the same as the union of four curves at time 0. Note that there are actually a couple of variations of the tile; the tile consists of a central ⊓ shape, with added "legs" connecting this shape to the other tiles. This shape is shown in Figure 2, which shows a quarter of the tile compared with $t = 0$ in Figure 1.



$t = 0$ $t = 0.25$ $t = 0.5$ $t = 0.75$ $t = 1$

$t = 1$ $t = 1.25$ $t = 1.5$ $t = 1.75$ $t = 2$

Figure 1: *Elaboration on Figure 3 (a) [V]*

Figure 2 (b) [V] shows several different levels of iteration in the same figure, namely, with values

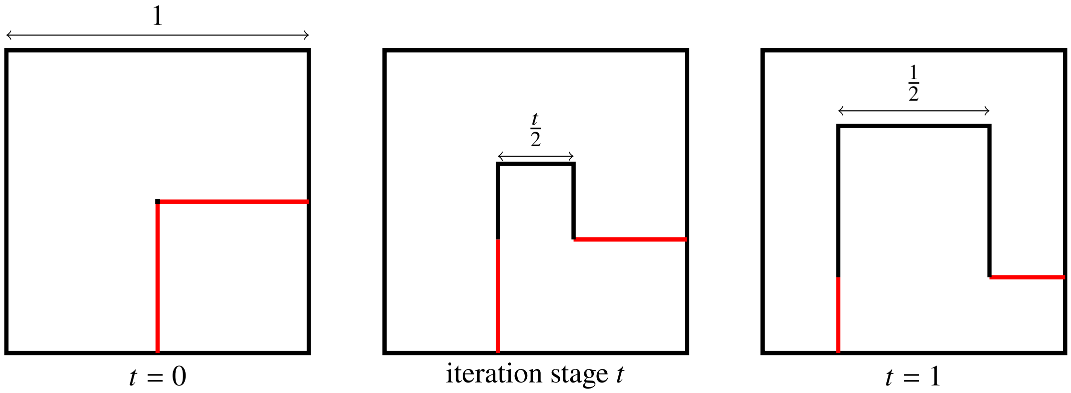


Figure 2: *Lengths of components in deformation. The central shape increases from a point to a shape consisting of three sides of a square, the side length of which is at the final stage half the length of the tile side length. The red lines are added "legs" which will be added in different directions depending on the tile. The tile "legs" and tile orientation are determined recursively depending on the tile which this tile is replacing, and its position within that tile, and that tile's orientation.*

$t = 0, 0.5, 1, 1.5, 2, 2.5, 3$. Figure 3 shows these 7 different stages with constant values for t across the image, from which these subtiles in Figure 2 (b) [V] are taken. In Figure 4, the image is further divided, into tiles where the iteration level is given by $t = 0, 0.25, 0.5, 0.75, \ldots, 3.75$, with the same background colour rule. By varying the value of t continuously across the image, we would end up with a smooth curve, without the discontinuities seen in this figure. The smooth case is just a limiting case of enough divisions being taken that the result looks smooth.

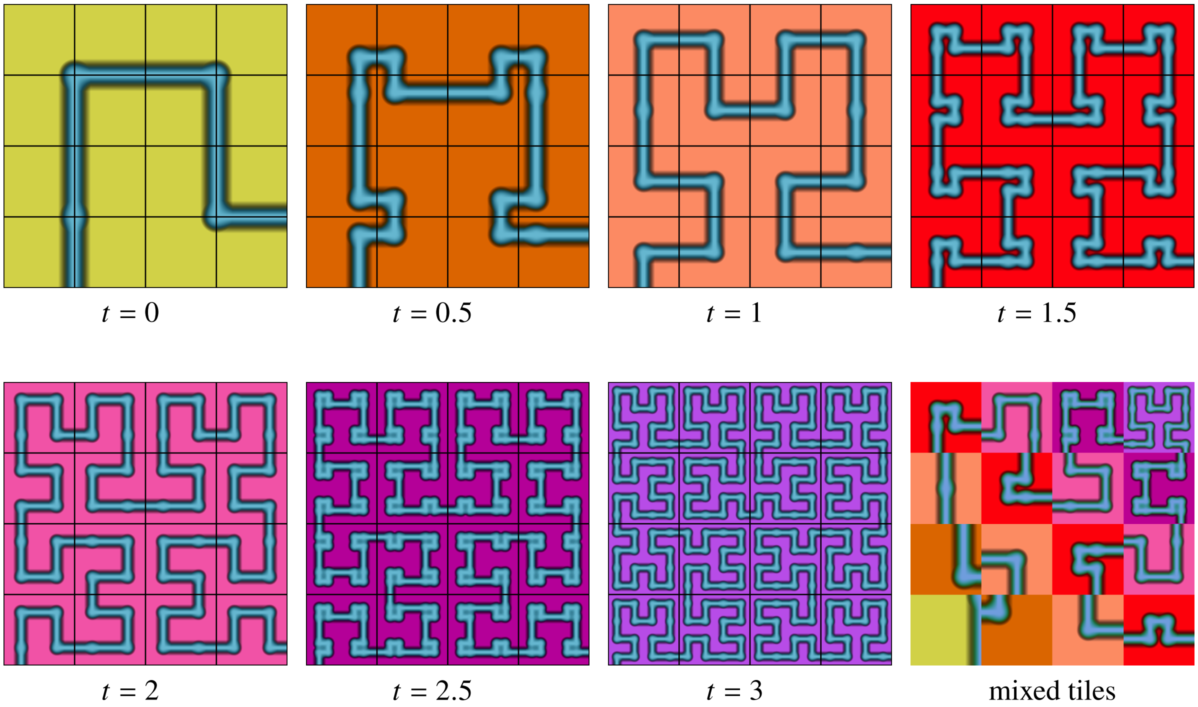
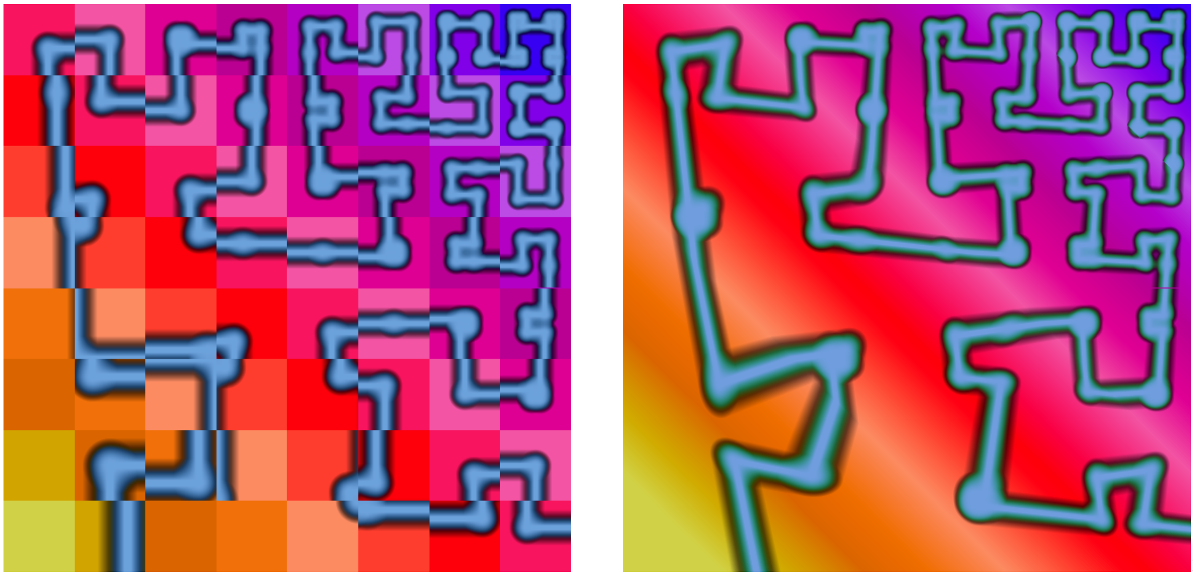


Figure 3: *Elaboration on Figure 3 (b) [V]*



;

Figure 4: *Different parts of the left image have values $t = \frac{i}{4}$ for $i = 0, 1, \ldots, 14$. On the right the progression is continuous.*

In Figure 3 and 4 the value of t is an approximation to $t = t(x, y) = 2(x + y)$, for $x, y \in [0, 1]^2$, but any other positive real valued function could be used. Figure 3 (d) [V] uses $t(x, y) = t_d(x, y)$ where t_d is a function given by $t_d(x, y) := 2(x + y) - 0.7$ in order to have a smaller value for the difference $|t_d - d_b|$ where $t_b(x, y) = (\lfloor 4x \rfloor + \lfloor 4y \rfloor)/2)$ is the function which is used in creating Figure 3 (b).

In Figures 3 (b), (c), (d) in [V], the values of t in terms of the coordinate position (x, y) with $0 \le x, y \le 1$ are given by functions t_b, t_c, t_d respectively, of the form

$$t(x, y) = g(x) + g(y) + c$$

for some constant c. The constant c is given by $0, -1, -0.7$ respectively. This choice of functions is used since they have a similar form and so the transition from the discrete changes in (b) to the change of constant rate of change in (d) can be more easily compared.

The function $t(x, y)$ is given respectively by

$$g(x) = \begin{cases} \frac{1}{2}\lfloor 4x \rfloor & \text{case (b)} \\ \frac{1}{2}f(4x) & \text{case (c)} \\ 2x & \text{case (d)} \end{cases}$$

where $f(x) = \lfloor x \rfloor + \min(1, 3\{x\})$, where $\lfloor x \rfloor$ is the floor function, and $\{x\}$ is the fractional part function. A graph of $f(x)$ is shown in Figure 5 (a). Figure 5 (b) shows $g(x)$ in the three cases. Figure 5 (c) shows $t_b(x, 0)$, $t_c(x, 0)$, and $t_d(x, 0)$. (Note, since I have not defined a negative iteration, any negative values of these functions are changed to 0.)

Note that f is a member of a family of functions given by $f_\lambda(x) = \lfloor x \rfloor + \min(1, \lambda\{x\})$ for a constant $\lambda \ge 1$, which interpolates "smoothly" between x and $\lfloor x + 1 \rfloor$ as λ goes from 1 to ∞. When $\lambda = 1$, we have $x = f_1(x) = \lfloor x \rfloor + \{x\}$, and the larger λ is, the closer f_λ is to $\lfloor x + 1 \rfloor$. I put smoothly in quotation marks, since the limit of this sequence of continuous functions is discontinuous; We have a convergence,

$$\lim_{\lambda \to \infty} f_\lambda(x) = \lfloor x + 1 \rfloor,$$

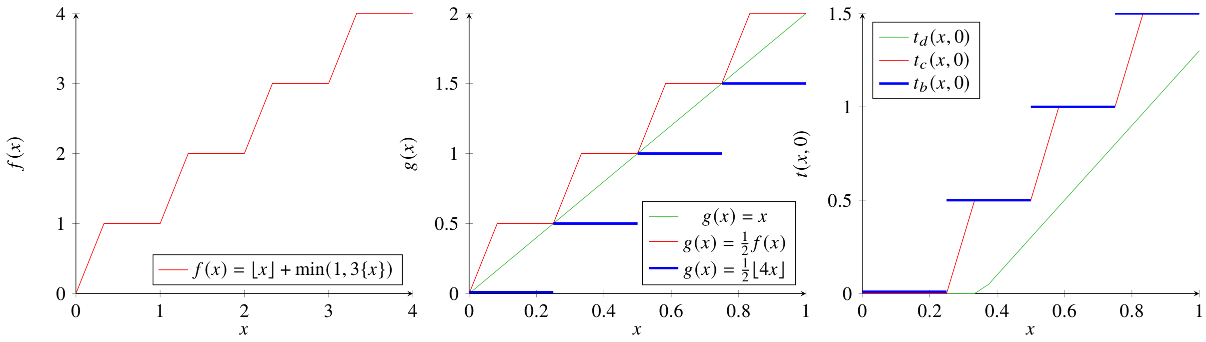
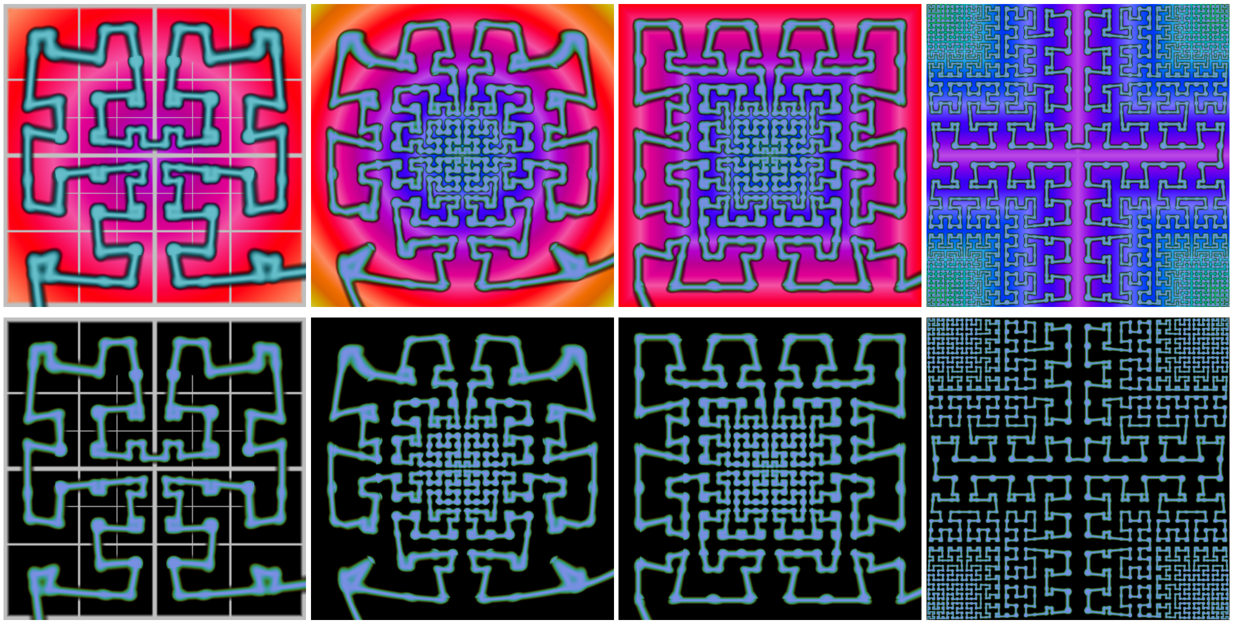


Figure 5: *Graphs of functions related to determination of iteration level.*

but we do not have *uniform* convergence.

Some examples of images created with different functions $t(x, y)$ are shown in Figure 6 which uses the same value colour scale as in Figure 3 [V].



$3 - 3||(x, y)||$ $5 - 7||(x, y)||$ $5 - 7||(x, y)||_\infty$ $6 - 6||\mathbf{m} - |(x, y)|||_\infty$

Figure 6: *Examples of different variations of iteration level* $t(x, y)$ *where* (x, y) *is the Cartesian coordinate of a point in the square in each case, with the squares having side length* 1 *and* $(0, 0)$ *in the centre. Therefore* $||(x, y)||$ *denotes the distance from the centre measured in the usual Euclidean way, and* $||(x, y)||_\infty$ *denotes the distance with respect to the* L_∞*, or "max" norm. The first row colours the background level according to the value of* $t(x, y)$*, for easy of seeing the iteration level, and the second row has the same* t *used for computing the curve iteration level, but black background, for greater clarity of the curve. In the left most images the tile boundaries are also shown. In the rightmost case* $\mathbf{m} = (0.5, 0.5)$.

In Figure 3 (e) [V] a more random function is used for t. These were all written in WebGL, which makes it particularly easy to vary a parameter across an image.

Figure 7(a) in [V] shows a version of the discussed hinged tiling fractal with varying iteration level across the image. Figure 8 shows a similar image with constant levels of iteration. Figure 9 shows a few different versions of this figure, with different backgrounds tiles for greater clarity. The constant level of iteration in Figure 8, are replaced with varying iteration level in Figure 9, as in the Hilbert curve case. Note that in the rightmost image of Figure 9, the boundaries of the tiles change in a discontinuous manner. The reason for this can be seen by comparing with the Hilbert version in Figure 1, where it can be seen that although the curve changes continuously, the boundary of the tiles (this is the division into square tiles, shown by the gray lines) only changes discretely, depending on $\lfloor t \rfloor$.

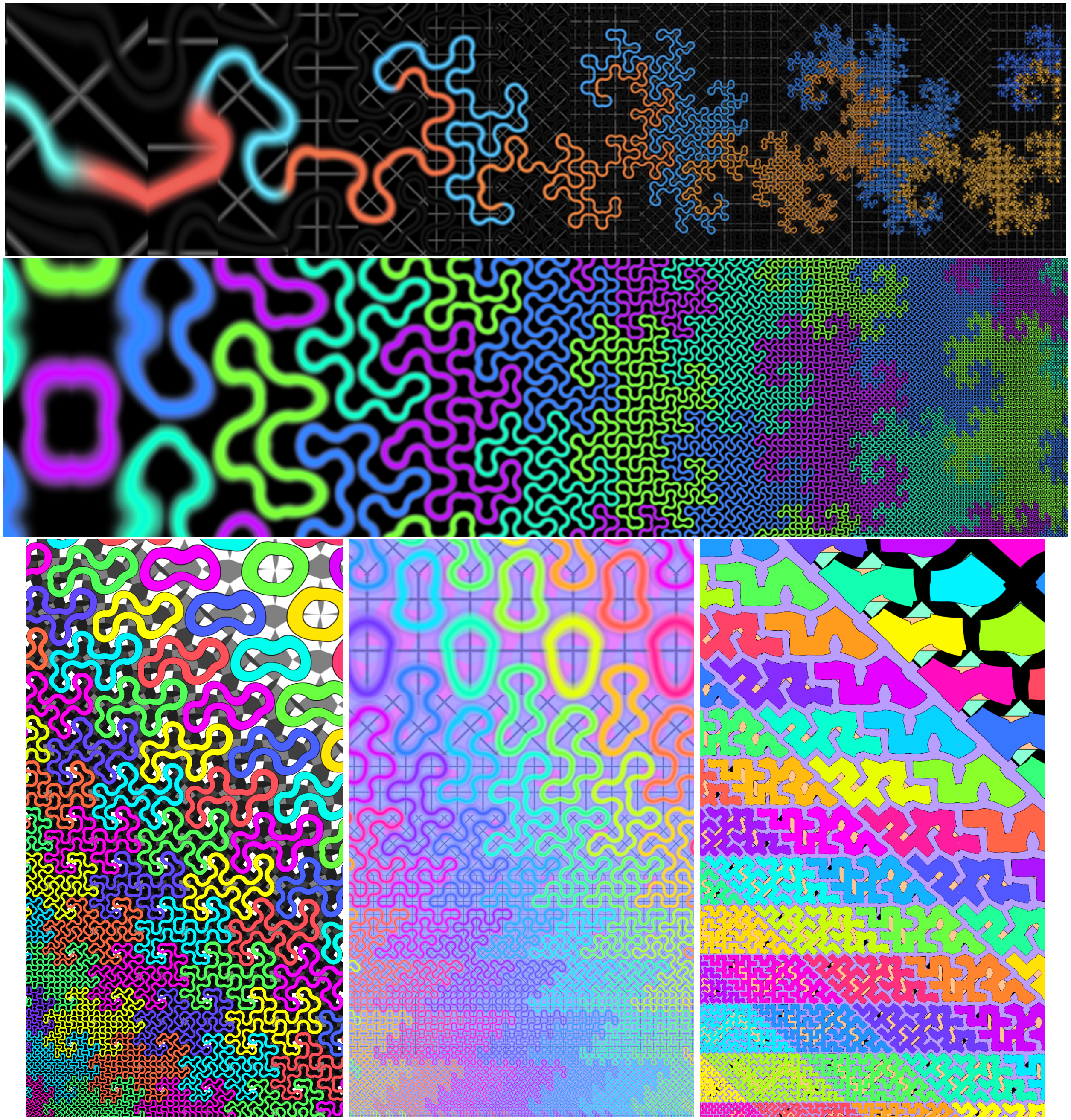


Figure 7: *Variations on Figure 1 [V].*

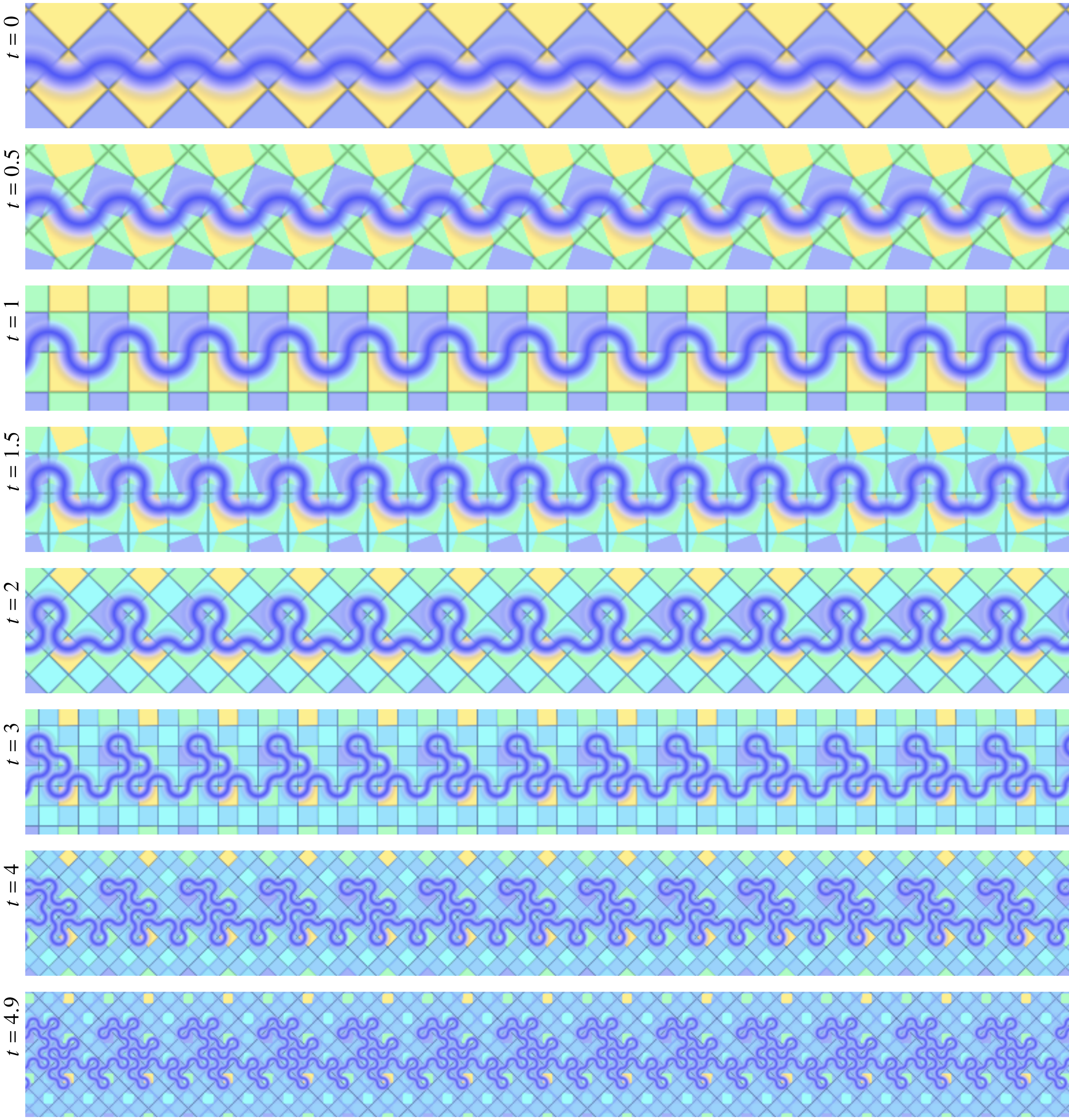


Figure 8: *Images similar to Figure 7 (a) [V], but with constant t per image, and background tiles.*

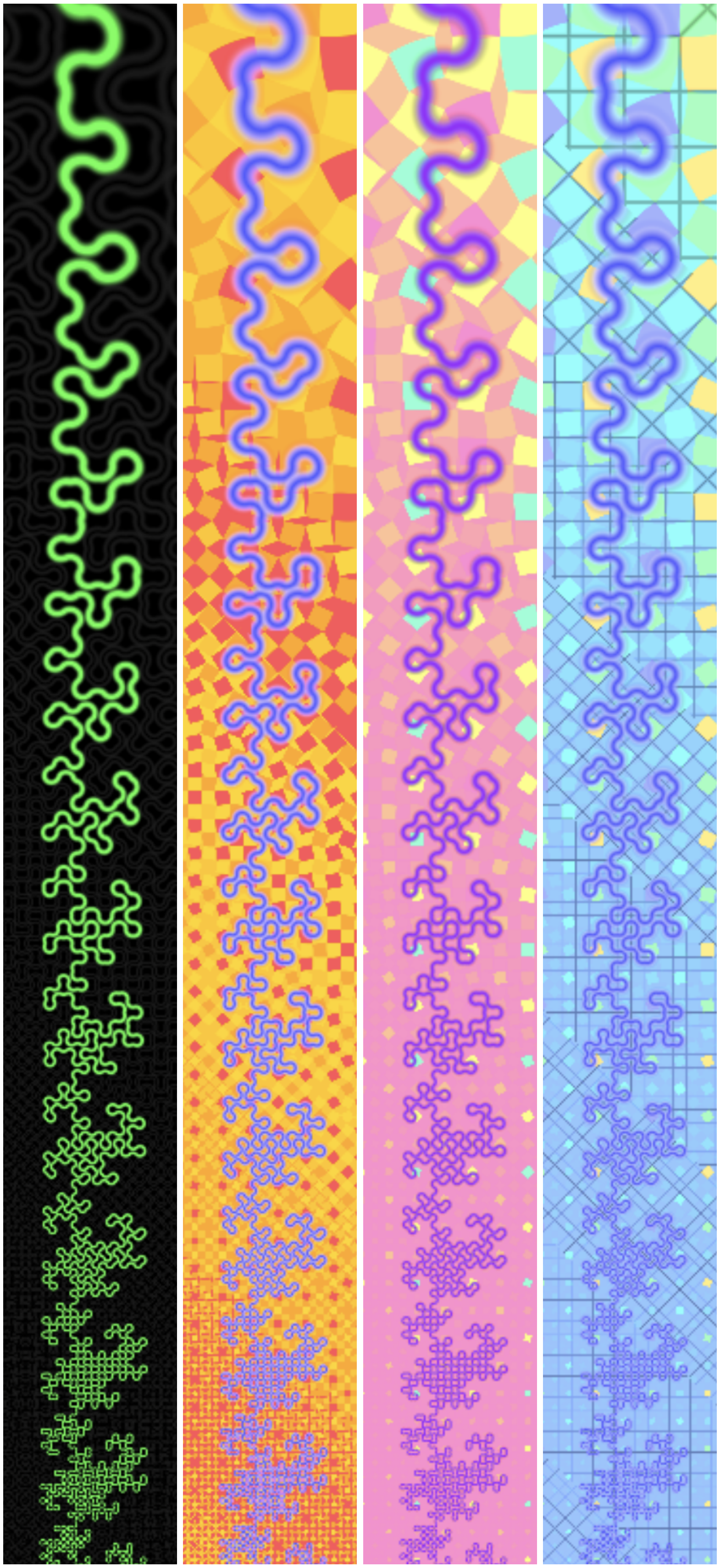


Figure 9: *Increasing iteration level down the page*

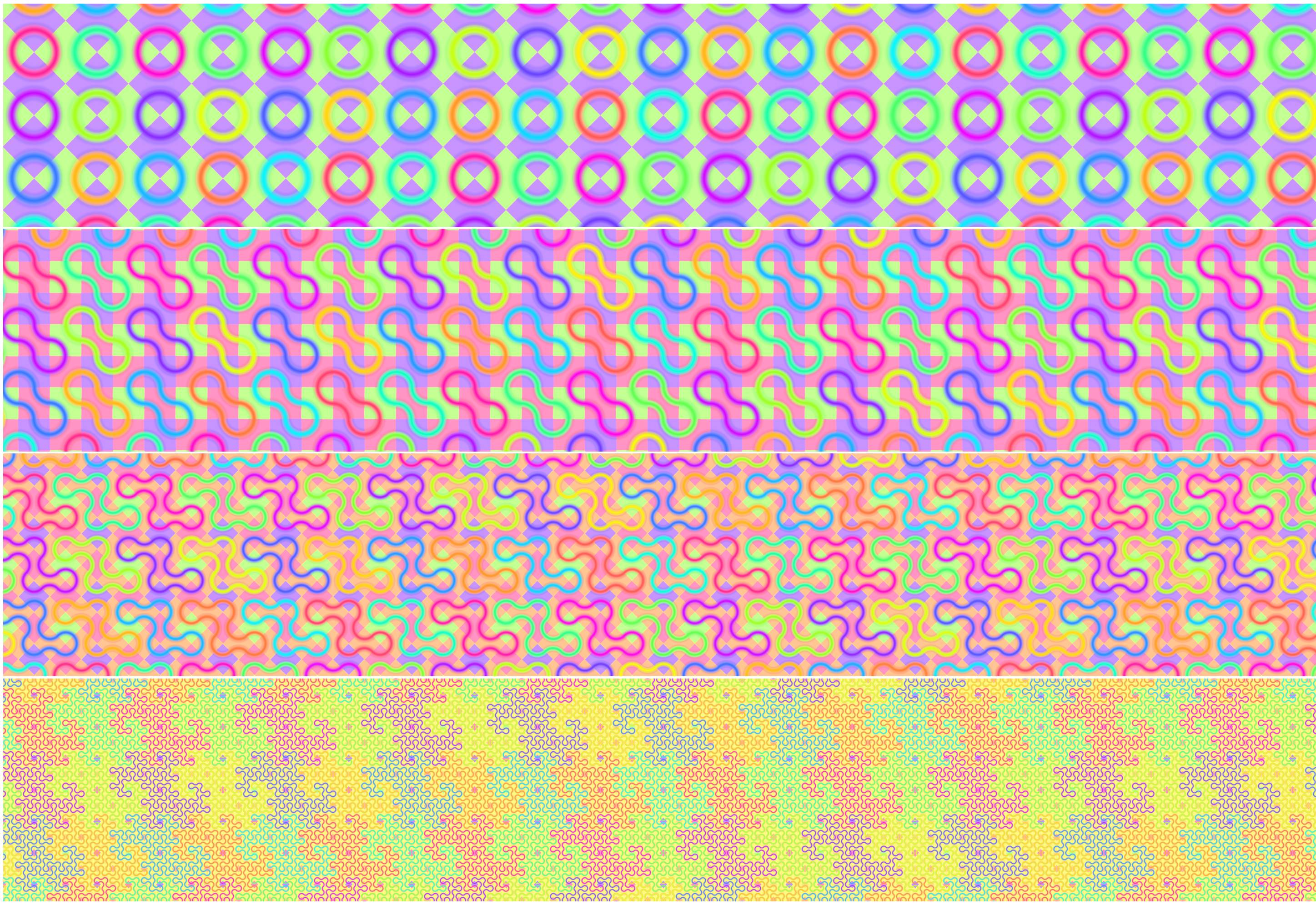


Figure 10: *Constant level t of an image similar to Figure 1 [V].*

Figure 1 [V] is obtained in a similar way. A few versions of constant level t images are shown in Figure 10, which correspond to different parts of Figure 1 [V], which has a varying value of t across the image. Figure 7 is created in a similar way, but with possibly different initial tile configuration, and different backgrounds, e.g., only showing tile edges.

In Figure 6 [V] two operations are shown corresponding to different directions of rotation of the hinged tiles. The effect of these on the curves will appear as a "squashing" in one of two possible directions. This can be seen in Figure 11 for various different examples.

Figure 7 (c) [V] shows an application of operations in the sequence 110110110, applied at a variable rate, with more iterations towards the centre of the image. Repeated infinity often, this sequence gives rise to one of the fractal dragon curves. The idea of the image Figure 7 (c) is to show the fractal dragon in the centre surrounded by a sequence of its approximations. Some other examples are shown in Figure 12.

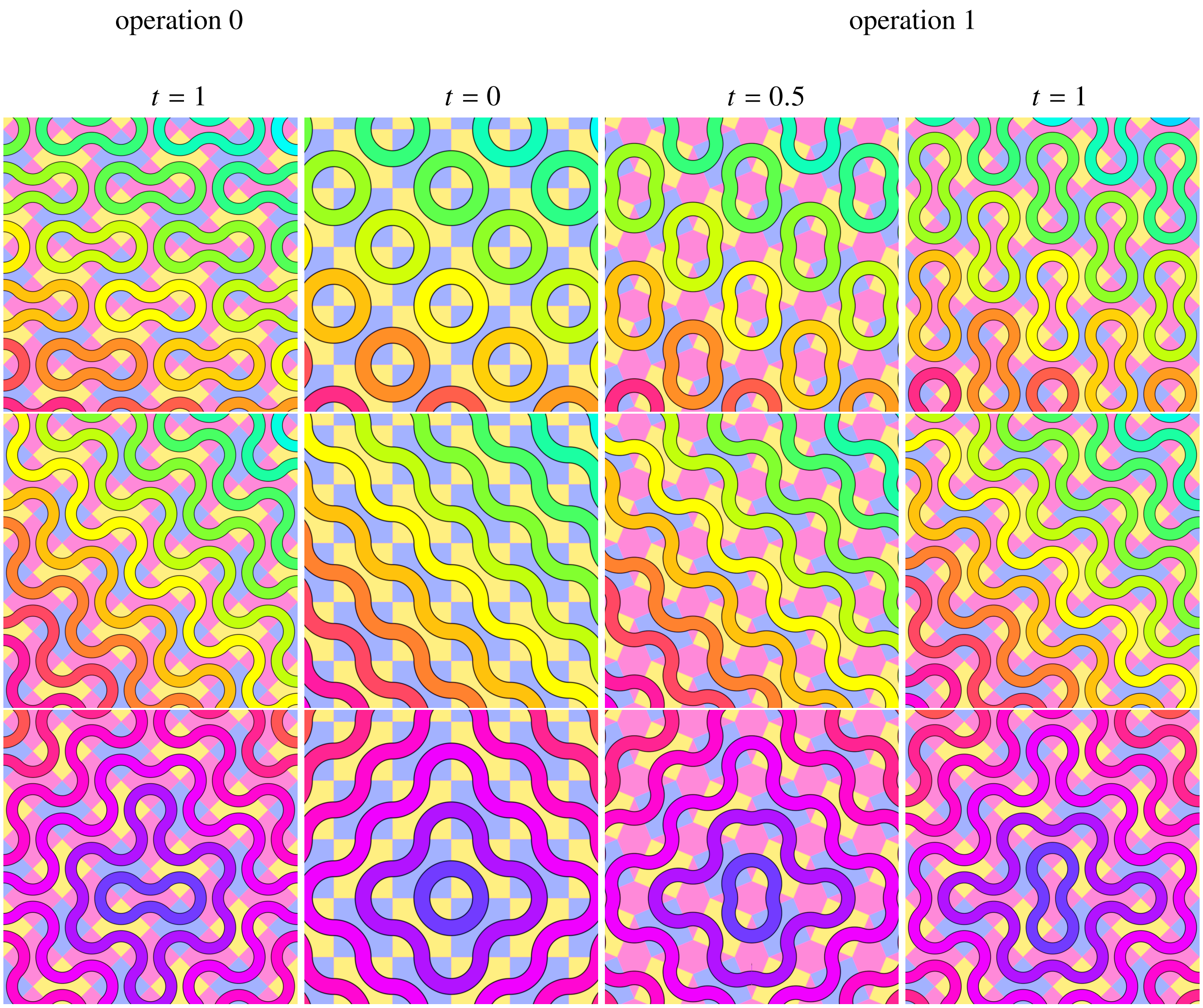


Figure 11: *Different operations on different configurations. The second image in each row is with $t = 0$; to the left and right we have application of operation 0 and 1 respectively. The first image in each row has $t = 1$, the third has $t = 0.5$ and the fourth has $t = 1$.*

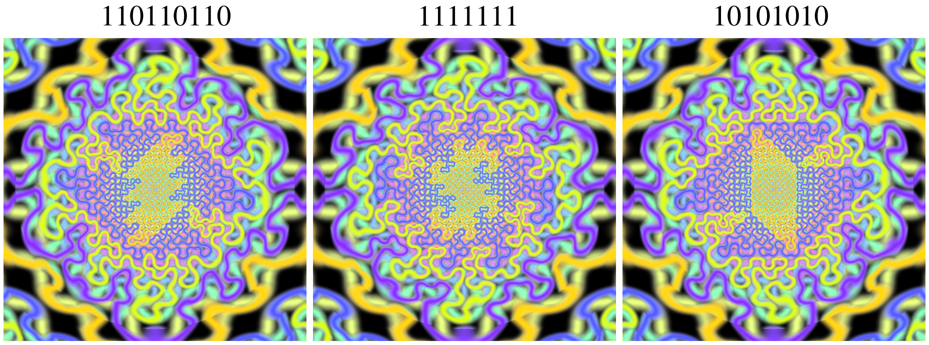


Figure 12: *Variant on Figure 7 (c) [V], different operation sequence.*

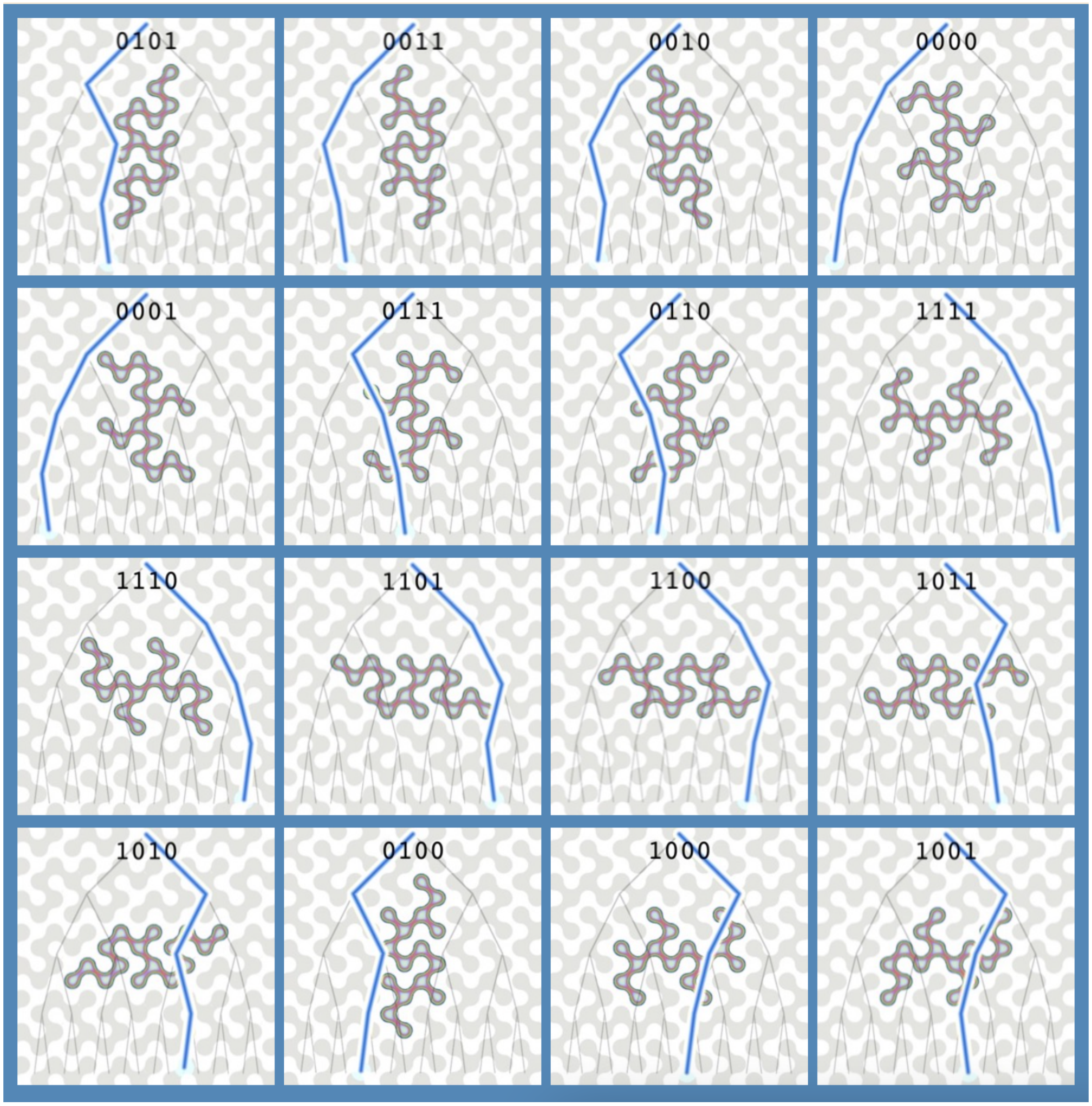


Figure 13: *16 possible ways to apply 4 hinged operations. The curve is superimposed with the corresponding path in the binary tree of choices of direction of hinging at each iteration step.*

Figure 13 illustrates how the 16 possible applications of 4 operations result in different curves. These can be thought of as paths in a binary tree, so the corresponding tree and path is shown. These images come from a JavaScript/WebGL version which can be continuously varied to pass via a continuous deformation from each of these configurations to the other.

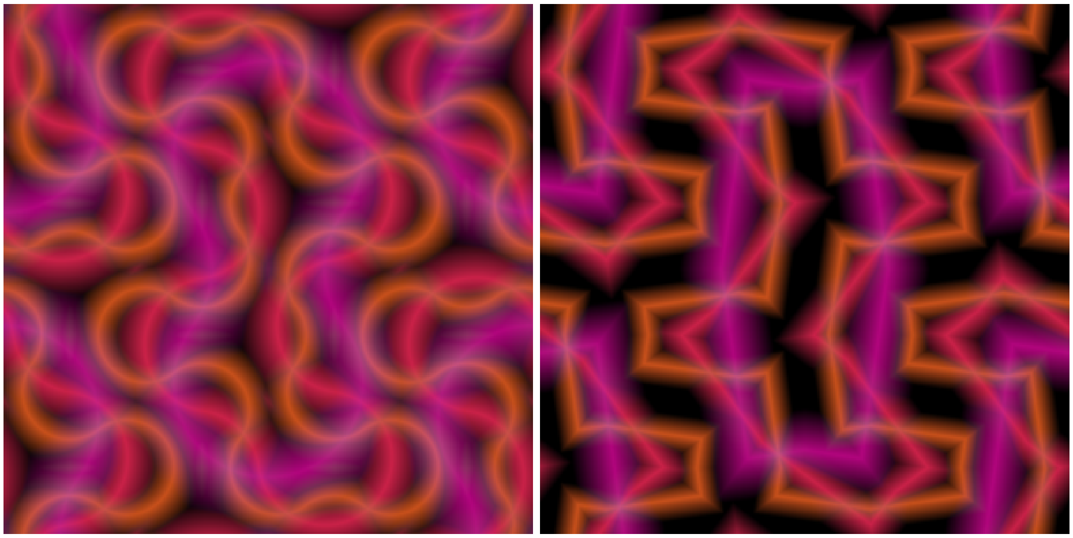


Figure 14: *Comparison of curved vs straight line paths on the tiles.*

There are many ways to vary these tiles, for example, the curves of the Smith Truchet tiling can be replaced by the right angle paths shown in Figure 4 (b) [V]. A few superimposed iterations can be seen in Figure 14 for comparison. Figure 15 shows the 8 tiles obtained from the application of three operations. The tile in this case is one with the quarter circle curve replaced by a straight line. This figure shows filled in vs unfilled versions, and black background vs showing underlying tiles. The focus of the paper is in particular on variation of the level of iteration and the hinged tiling method, which Figures 14 and 15 don't really use to a significant extent, but are included due to their close relationship to the other tilings. Figure 16 shows a varying version of the tiles, with operation 00000000.

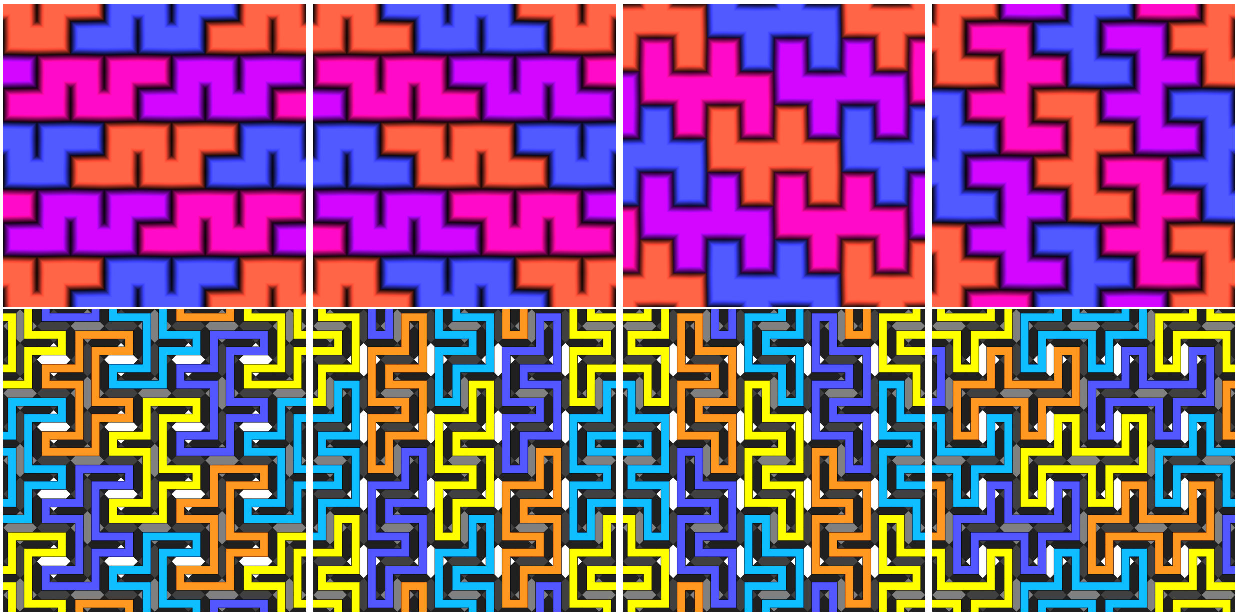


Figure 15: *Tilings from three operations.*

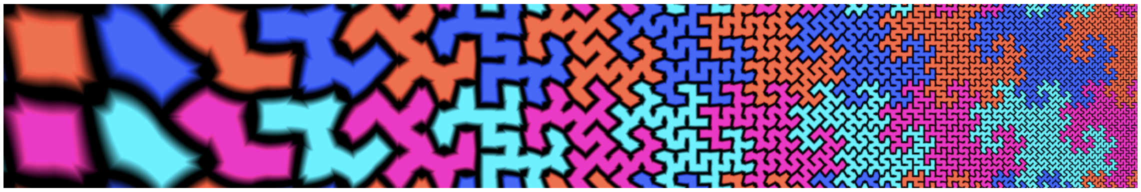


Figure 16: *Increasing level of iteration from left to right.*

In the varying Smith Truchet examples in Figure 9, the iteration level depending on the distance from the top of the image. This corresponds to an iteration level of the form $t(x, y) = y$, where y is the distance from the top of the image. The function $t(x, y)$ can be replaced by other functions, for example, as shown in Figures 17, 19 and 18.

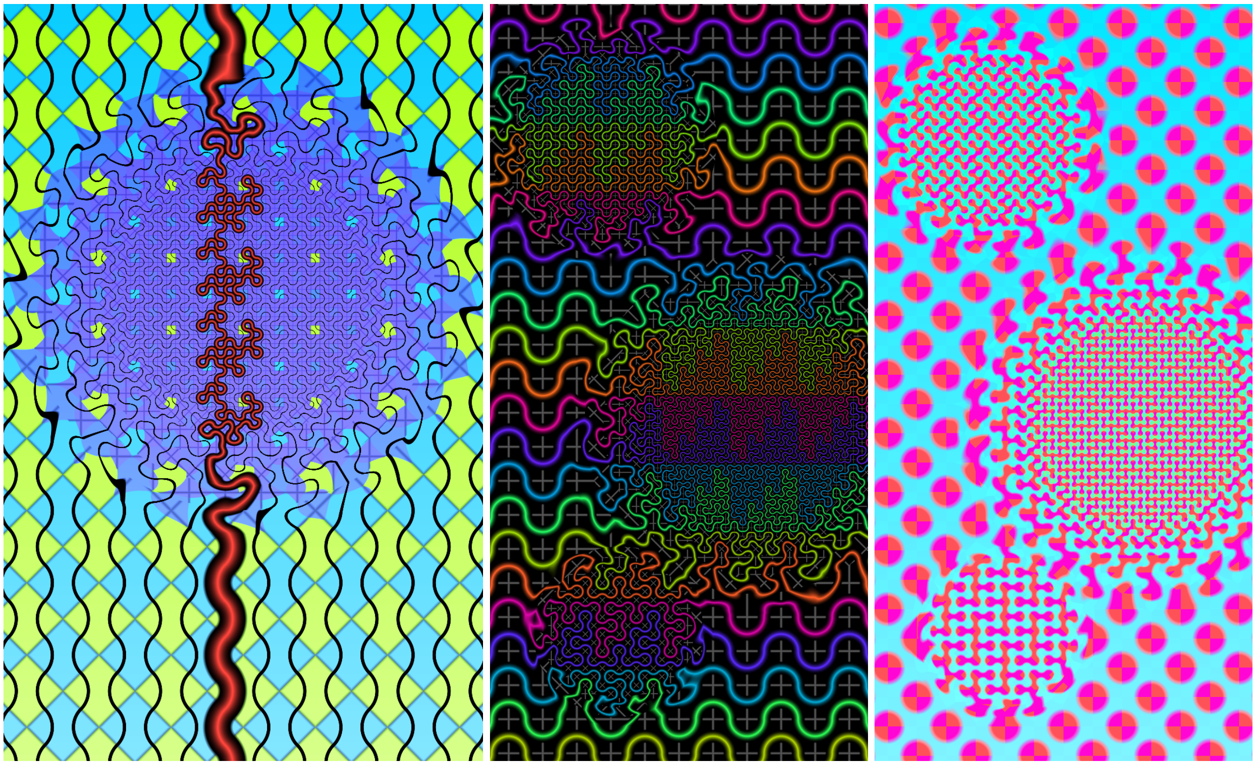


Figure 17: *Iteration increased in circles. The iteration level is constant within the circle, and constant sufficiently far outside the circle. Near the boundary there is a continuous change in variation between these two values.*

In these Figures 17, 19 and 18, $t(x, y)$ depends on the distance of a point from the centres of various circles. The same three circles can produce many different result depending on the other parameters chosen. This process could be applied to other shapes, to give an image filter. The rate of change between the foreground and background should be carefully adjusted, as too great a rate of change may result in too much distortion. Notice that in the iteration step, at each stage I have introduced a single new colour for the newly introduced squares. This means that at each unit increase in t, the 50% of the colour of the background tiling becomes the new tile colour. When the tiles are small enough, the new colour significantly predominates, adding to the image filter effect.

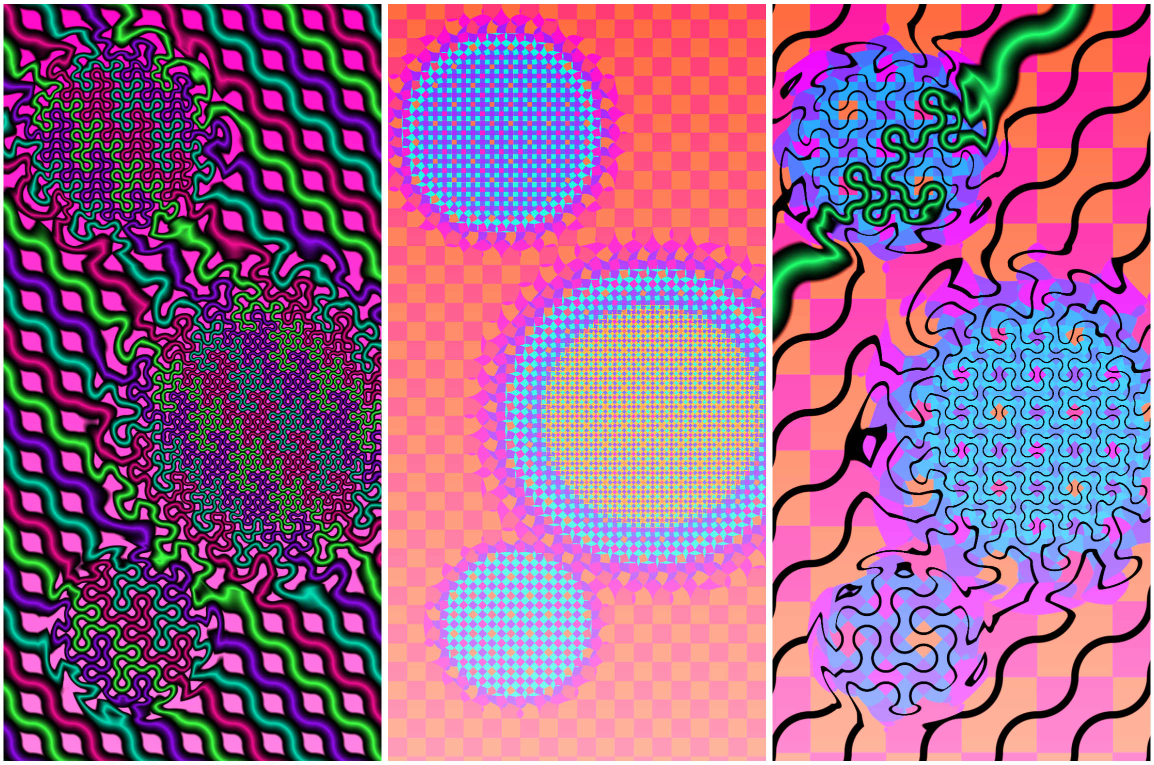


Figure 18: *Further examples of iteration increased in circles.*

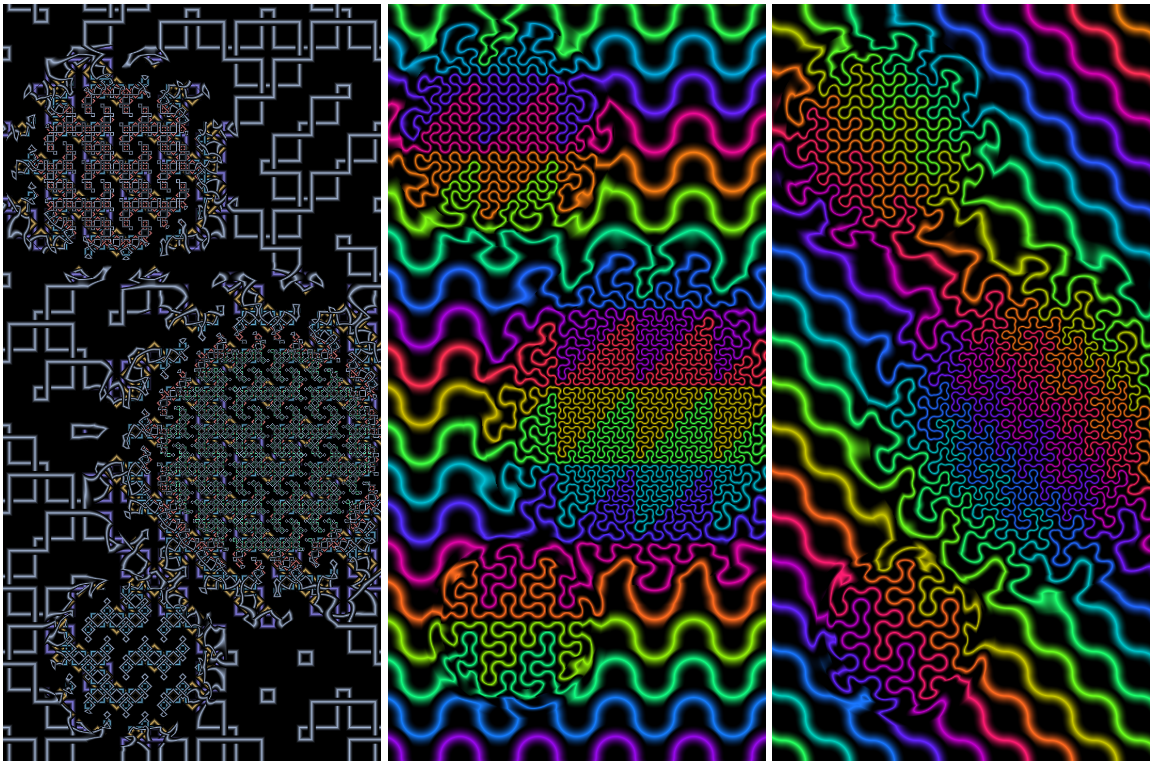


Figure 19: *Further examples of iteration increased in circles.*

I have written an image filter version of the hinged Truchet tiling method at https://www.mathamaze.co.uk/GA2021/index14.html, and some results created with that program are shown in Figure 20. That program did not incorporate background tile colouring, and various improvements made in later programs implementing the hinged tiling algorithm. However, the later do not currently have image upload facility implemented. The image filter could be developed further to produce better quality images.

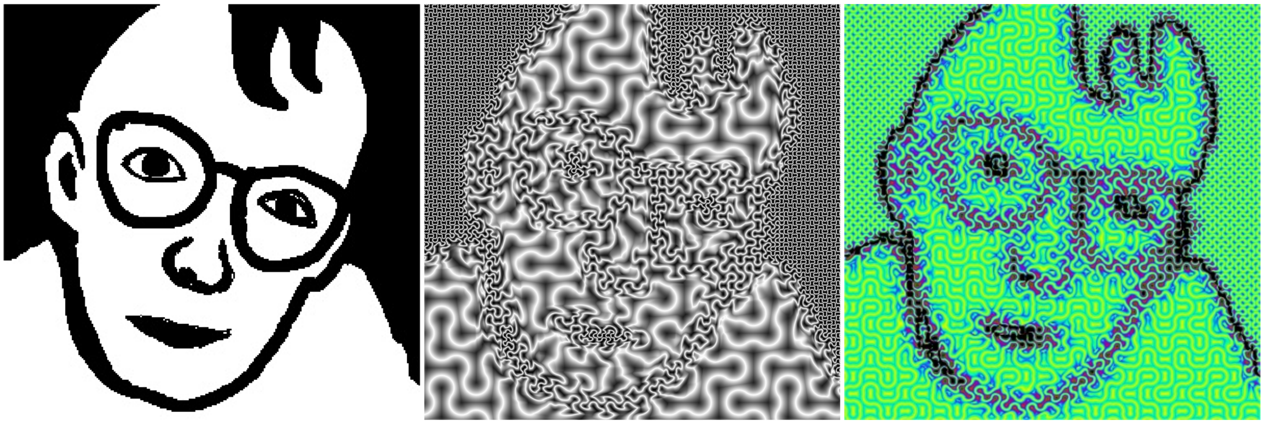
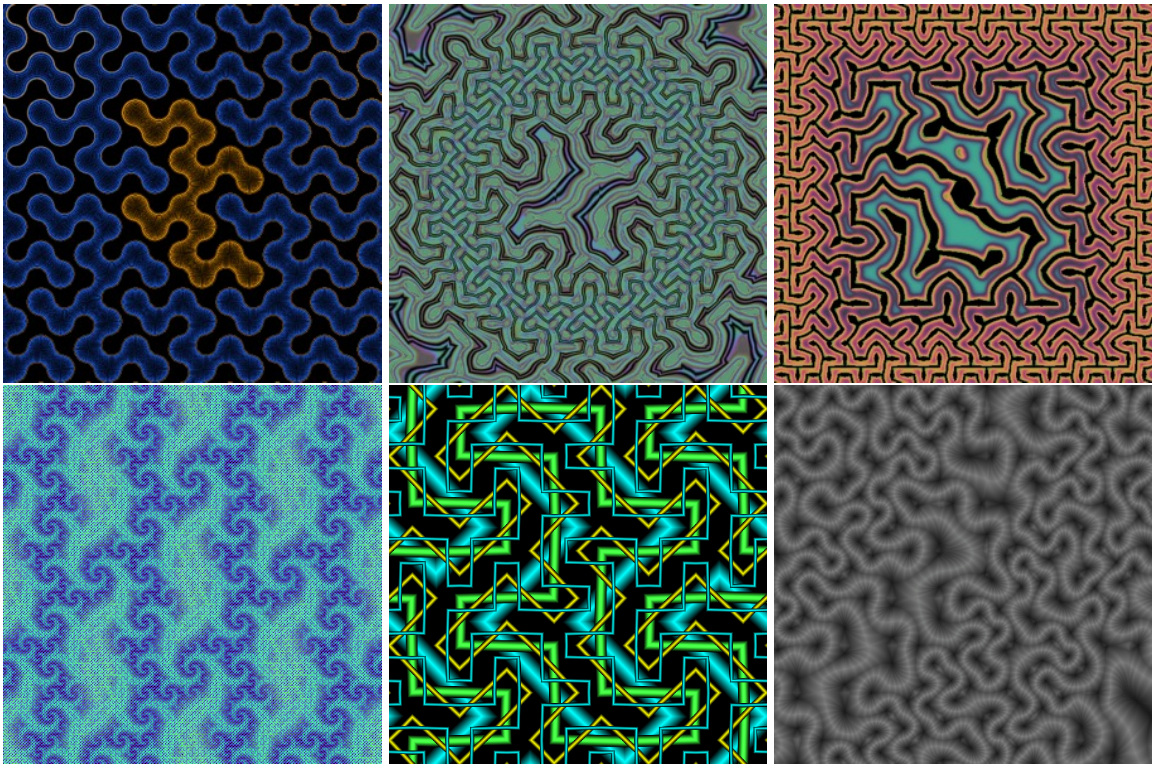


Figure 20: *Simple image filter, implemented at https://www.mathamaze.co.uk/GA2021/index14.html. The image on the left is the input, the other two images the output. This image filter is simply based on the stage of the iteration depending on the brightness of a pixel. It works best with simple black and white images. Details are hard to pick out in the middle image, so colour is added for the right image, though this may be considered "cheating".*



In the last paragraph of [V], I mention colouring problems; there are various colouring problems involved in creating these images. For the remainder of this note, I discuss the choice of background tile colour. In order to choose the colour of the background tiles, I assign background tiles introduced at integer stage n with a colour index n, i.e., a number to determine colour. Alternatively we could colour all tiles the same colour at stage zero, or randomly, or some other method, I am just describing my choice here. I wanted to have tiles introduced all in the same colour at each stage to give the impression that the hinging of the already produced tiling was taking place against a constant colour background. Hinged tilings with non constant patterns or even moving images are certainly possible, e.g., see https://www.mathamaze.co.uk/circles/p5js/ex4.html or https://www.mathamaze.co.uk/GA2021/index4.html. However, too much going on would obscure the fractal dragon curves, which are the focus here. As an alternative to "birth date", the tile colour can also be determined by powers of two dividing $i + j$ and i where (i, j) is the coordinate of the tile, as indicated in Figure 21.

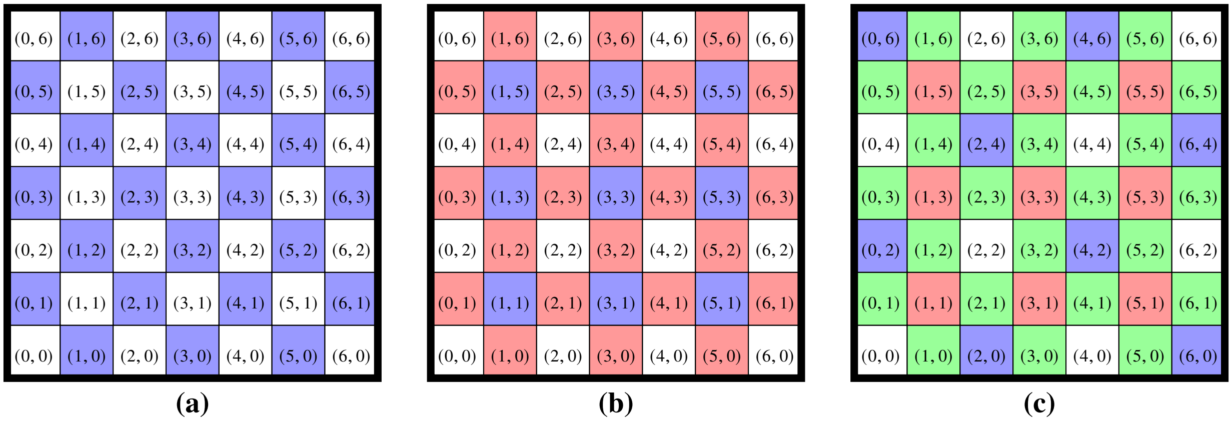


Figure 21: *Successive stages of iteration after new tiles added. These are all shown in the same orientation, rather than every other set of tiles being at* 45° *degrees rotated, for convenience in this drawing. This means the images don't correspond exactly to the program, but they illustrate a colour index system. Tiles are reindexed after each operation, also not necessarily exactly as in this diagram in the program. In (a), the colour depends on* $i + j \mod 2$. *In (b), the colour also depends on* $i \mod 2$. *In (c), the colour also depends on* $i + j \mod 4$. *The order of colours added in this sequence is white (initial colour, corresponding to a not shown zero position of only white squares), blue, red, green.*

Note that the tiles have to be carefully relabeled at each step in order to have compatibility at each iteration, since I don't want tiles to change colour in any kind of abrupt way. Once a colour number is assigned to each tile, another function is used to determine an appropriate colour. Note that at each stage, the new tiles touch all of the previously created tiles. Therefore, it is difficult to choose a sequence of colours such that each new colour is a distinct different colour from all the previously introduced colours. Also, when the tiles are very small, since the new tiles introduced take up 50% of the area, the new colour will predominate. For example, in Figure 22 the colour is given by a function of the form "colour of tile with colour index n is equal to colour $n \mod 4$", so there are just four colours, and the predominant colour can be seen in the striped appearance of the image, where the iteration level depends only on the distance from the left side.

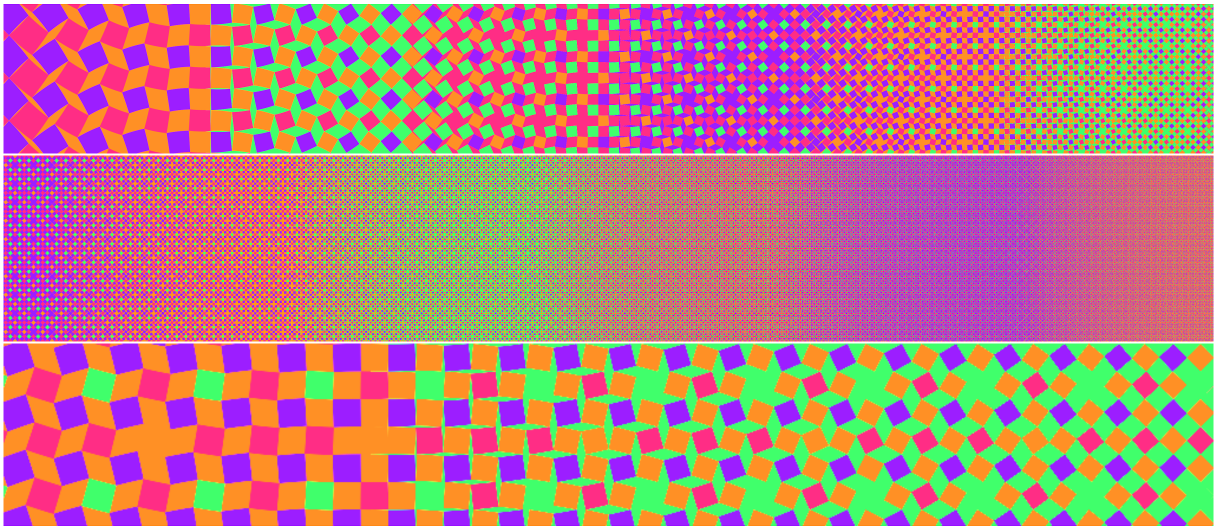


Figure 22: *Iteration level increases linearly with distance from left to right. At each stage, the new tiles are all given the same colour, which is cyclically either red, purple, orange, or green. The first row shows iteration levels passing from level* 0 *to* 6*. The second shows iteration levels passing from level* 4 *to* 9*. The third row shows the completion of adding orange tiles at iteration step* 6*, which will make orange the predominant colour, and green least used, and the progression from level* 6 *to* 7*, where green tiles are reintroduced and again green becomes the dominant colour. If green was never reintroduced, eventually the green tiles would be a negligible area of the tiling. Notice that since each time a new tile is introduced it touches tiles of all previously introduced colours, we can not avoid having orange tiles touching orange tiles, or green tiles touching green tiles, etc, giving rise to the cross shaped configurations in the image.*

Further space filling images: