Articles

INFINITE TILES OF REGULAR REP-TILES

Tony Hanmer

Georgia Today
a.hanmer@gmail.com

Dedicated with huge gratitude to N.W.-B., who introduced me to fractal programming and thus gave me the main tool with which to soar through endless multiverses.

Abstract: *Here I describe an infinite number of fractal tiles of regular rep-tiles in all dimensions above 1. Each rep-tile's set of tiles can be divided into subsets based on certain visual characteristics. As fractals, they can be programmed and rendered in any size. They can be arranged in groups according to their aesthetic properties; used as an unending visual and pattern-recognition training ground for AI; and even animated as increments from one to the next.*

Keywords: rep-tiles, fractal, tiles, square, triangle.

Preliminary Note

Should I apologize for including so many large pictures in this article? There are an infinite number available, each with unending detail, and I have found an infinitesimal 10000 or so of these; so quite a few deserve to be displayed to give some idea of the endless diversity available.

The nucleus of this article originally appeared in two articles in *Georgia Today* in November 2017 [3]. It has been hugely updated and edited for this new version.

Origins

In 2000, I was programming new fractals using the MS-DOS-based freeware programme, FractInt [2], which offers a huge set of fractal types to explore,

zoom into, colour and program from scratch, and is also now available for Windows. Although I read and experimented all I could, many of these fractals remained fairly opaque to me as far as their mathematical workings were concerned. The types that I was experimenting with, however, are about as simple a fractal as one can make, and these made a more suitable starting point for me.

One of my early chance discoveries was a tile of a square of side length 3, a great delight to find. This was followed by another tile, of a side length 2 equilateral triangle, then another square tile, and in great excitement I began looking for patterns and rules in these items and ways of finding more. These took time to tease out. The earlier tiles are obvious as tiles, and my innate pattern sensitivity enabled me to recognize them, thrilling and spurring me on; but finding the governing construction principles, as well as many subsets along the way, took time and patience, with periods of feverish activity at the computer, programming mixed with time away in contemplation and making notes by hand. This dual-approach method seems to work best for me in these situations. Intuitions followed by experiments. Often a flash of understanding would come when I was away from what then was a desktop machine, and I had to document it on paper before testing it out and either running with it into new territory or abandoning/tweaking it. I would also ask to upload the freeware FractInt programme to friends' machines if my absence from home was prolonged and inspiration burning in me, and continue thus, becoming one of the programme's "agents of reproduction"...

Figure 1: My 1st ever tile find, $\frac{1}{4}$ of a square of side length 3. At the limit it consists of an infinite number of pieces joined by what I call "wasp waists", or mathematical points.

Figure 2: My 2nd tile find and 1st of a triangle, $\frac{1}{3}$ of the triangle's area and of side length 2. At the limit it consists of an infinite number of pieces joined by what I call "wasp waists", or mathematical points.

Figure 3: My 3rd tile find and 2nd of a square, $\frac{1}{4}$ of its area and of side length 3. At the limit it consists of an infinite number of pieces joined by what I call "wasp waists", or mathematical points.

When finally I had the production rules more or less nailed down, I remember noting in my journal: "This is the ONE kind of thing I would like to have found in all of mathematics." Because it's beautiful, infinite, and programmable. There was wonder, and gratitude. Being sensitive to patterns had just paid off more than I could imagine.

There are actually several different categories of tile of the regular reptiles, or rep-tiles¹. The "standard" ones fill the triangle in 3 pieces, the square in 4, so they need the same number of pieces as the rep-tile has sides. Variations fill both rep-tiles with a "dust" made of an infinite number of pieces at the limit; another type fills the square in 2 pieces instead of 4. Yet another fills the triangle in 6 pieces, and the square in 8. Each of these variations comes with an infinite number of examples.

¹A regular reptile or rep-tile 1) has all sides and angles congruent and 2) can be divided into smaller copies of itself endlessly. The regular rep-tiles in two dimensions are the equilateral triangle and the square; in three dimensions, the cube; in four, the hypercube. We will concern ourselves here mostly with regular two-dimensional rep-tiles.

Figure 4: Two examples of a tile of a square needing only 2 pieces, side lengths 3 and 5; a tile needing 8 pieces to fill the square; three “standard” (4-colour) square tiles.

As the tiles are fractals, they are rendered at the desired size by re-iterating them over and over until they become filled in completely at the desired size. There must be no overlap or holes when all the tiles are together, but a smooth fill, although this may be coloured or textured in various ways. Any individual tile, however, may have holes if these will be filled exactly by separate pieces of its neighbour tiles. The main two ways of rendering them so far are as L-systems (line-based fractals named after their inventor, Aristid Lindenmayer [4], or as IFS ². I prefer to code my L-systems in FractInt, and made most of my original IFS in Brazil IFS Generator ³, although this latter programme is now hard to find online and to use in current Windows versions. Indeed, most true IFS programmes have been superseded by Fractal Flame versions, related but different, such as Apophysis [1].

In any way of rendering⁴, a tile has two functionally distinct but otherwise identical sets of parts (hereafter called holons⁵), exterior and interior. The rep-tile's side length is divided into smaller copies of itself, numbering the square of this side length, and these are further broken into their own perimeter pieces which are straight line segments, and which become the holons. So a size 2 triangle has 4 smaller triangles making it up, as does the size 2 square. These copies share pairs of overlapping *interior* line segments, which are thus all double-sided lines, while the *exterior* ones are all single-sided. A size 2 triangle thus has 2 exterior lines (holons) on each side, making 6, and 6 interior holons as well. A size 2 square has 8 and 8 total. From this size upward, interior holons number more than exterior ones for both 2d rep-tiles. The ratio of exterior to interior holons is always 1:(size-1). Each iteration squares the number of holons, and thus the detail.

Figure 5: Iterations 1, 2 and 3 of a square tile with all holons parallel, thus occupying $\frac{1}{2}$ of the square instead of $\frac{1}{4}$. The tile's basic form is already clearly visible by iteration 3.

²Iterated function systems (IFSs) are a method of constructing fractals; see https://en.wikipedia.org/wiki/Iterated_function_system

³<https://app.box.com/s/31kl6xfmq2bgbwvqeon7>

⁴It is also quite possible to render these 2d tiles “by hand”, an iteration at a time, in either a bitmap or a vector editing programme. One simply starts with the initial set of square or triangular holons for a particular tile, and reiterates them by replacing each holon with the whole set shrunk down to fit the size of 1 holon. Rinse, repeat for as many iterations are desired. In bitmap programmes, the exponential growth of detail will not add hugely to the size of the output file. In vector programmes, however, unless one can reduce complexity after each iteration by removing all interior detail, the file size does grow huge fast. I mention these rendering possibilities merely as curiosities.

⁵A *holon* is something that is simultaneously a whole and a part; see [https://en.wikipedia.org/wiki/Holon_\(philosophy\)](https://en.wikipedia.org/wiki/Holon_(philosophy))

Once I realised that merely using the right number of holons once, both exterior and interior ones, would lead to a tile occupying $1/(\text{number of sides of the rep-tile})$, I was free to explore the sets of possible tiles for any rep-tile, limited only by programming string length and render size limitations in FractInt and by computing power. The totals turned out to be much vaster for any side length than I could have imagined, however, even size 2, and also grow huge very fast as side length increases.

The only regular rep-tile in 3 dimensions is the cube, breaking into $2^3 = 8$ smaller copies of itself, the “cubed” because we are in the 3rd dimension. I have not yet even begun to investigate the possibilities in this space, staying with explorations in 2d. Although there are programs for making fractals in 3d too, such as XenoDream ⁶, it is likely that tiles of the cube will not intuitively be as obvious to the eye as those in 2d, although the programming, extended from 2d to 3, is straightforward. This might be because 3d tiles are not something we are used to seeing, and might also be because there are 6 “standard” tiles of the cube, 1 per face, which is a lot for the eye to keep track of, even compared to 4 tiles for a square.

As for rep-tiles in 1 dimension... all we have is a straight line. It can be divisible into smaller copies of itself in an infinite number of ways, indeed, and a “smaller” infinity of these would be “tiles”, 2 of which would “fill” the line. But the only interesting shapes likely to result are if we make a 2d shape such as a square by placing different sets of these “tiles” one beside the other, making a

⁶L-systems and IFS share much, but not all, of their possibilities and characteristics for programming fractals. And there are important differences between them in rendering fractals too. L-systems renders use lines of perhaps small but nonetheless non-zero thickness. As a result, they “overstep” by a tiny amount in order that the resulting image file, be it a vector type (lines as output of mathematical expressions) or a bitmap (lines as sets of dots), contain lines which are visible at all. This may be fine for non-tiling images; but for tiles it means that there is a small but significant overlap between tiles, making them slightly larger than the true, exact-fitting tiles which they are supposed to be. The overstep does, however, decrease in importance and effect as the render size increases, and my current standard render size in FractInt is 30000×30000 pixels, making the overstep suitably tiny. Larger sizes can be rendered in this programme too, but this is the maximum pixel size for a lot of image editing programmes to handle. For the edge overstep reason, IFS fractals are a better choice of render, as their output may be considered shapes with a fill but no edge line thickness at all. However, programming IFS has its own challenges: 1) Making holons of a size which does not divide 100% (the whole IFS surface) evenly, i.e. dividing it by 3, 6, 7, 8, 9, 11, etc., means that the holons are not exactly sized for these side lengths, unless one can allow enough decimal points for the output at the chosen render size to be indistinguishable from exact. 2) In most if not all of the few existing true IFS programming systems, as opposed to related and more popular but different fractal flame systems, the possibility does not exist to “snap” a holon’s edge or corner to those of another holon, resulting in an exact touch between them without overlapping. This is especially important for holons of tiles of the triangle, less so for those of the square, for which edge and corner positions can be deduced without using trigonometry. The first challenge could be overcome by requiring holon sizes to be considered as fractions of 100, such as $1/3$, $1/6$, etc., and allowing the decimal expressions of these fractions to become as large as necessary for exact enough renders at the desired size. The second challenge merely requires building a “snap to” feature into programming, whether it be IFS or flame fractals (the latter’s pretty but unnecessary colouring algorithms perhaps being done away with by using colour files consisting of a single colour instead of a colour range and resulting in a solid-filled tile). Another possible way of getting perfect IFS tiles is to perfect the translation of their L-system versions into IFS notation, which should be fairly trivial. The L-system versions are an error-free starting point, and their corresponding IFS versions can be the required error-free renders.

pattern of stripes. We might then extend this idea into 3d, making a patterned cube of squares. I have not yet begun to explore the possibilities here.

Numbers

The calculation of *total tiles* of a particular rep-tile and size is a multiplication of the total (T) combinations (C) of its exterior (e) and interior (i) holons. Written mathematically: $T = C(e) \times C(i)$. That is all. Combinations and not permutations, because order of holons does not matter in the tiles, which are all 1-colour. There are duplicate tiles in each size, however, which we will address shortly. We start our calculations of totals with side length 2 because a unit-size triangle or square, side length 1, has no interior holons at all to use in making up its tiles.

We calculate the *total* number of a rep-tile's *holons* simply as its number of sides times the square (because we are in the 2nd dimension) of its size or side length. We divide this total into total exterior and interior holons thus: exterior holons are size $\times$ number of sides; subtracting this from the total yields the remainder as interior holons. We then divide exterior and interior holon numbers by number of sides to arrive at the numbers of exterior and interior holons available and needed to make a tile. And it is the numbers of combinations of these available holons from the total holons which give us the total number of tiles to be found in a particular size of a rep-tile.

Some numbers:

The triangle's tile numbers grow a bit more slowly as its size increases than those of the square due to its having slightly less holons to play with, but still get huge fast. Size 2: 2 out of an available 6 exterior and 2 of 6 interior holons are used, equalizing 15×15 combinations = 225 tiles total, the only size in which exterior and interior holon numbers are the same. For size 3 and up, there are always more interior holons than exterior ones, because some of the triangles which make them up are entirely inside the rep-tile. Triangle, size 3: combinations of (3,9) exterior and of (6,18) interior, equalizing $84 \times 18564 = 1559376$ tiles. From here on I will use exponential notation to save space as the zeroes mount up. The numbers are still exact, despite using decimals, as I include every digit until the zeroes start.

Size 4: combinations of (4,12) and (12,36), giving $495 \times 1.2516777 \times 10^9 = 6.195804615 \times 10^9$ tiles.

Size 5: $C(5,15) \times C(20,60)$.

Size 6: $C(6,18) \times C(30,90)$.

Size 7: $C(7,21) \times C(42,126)$.

Size 8: $C(8,24) \times C(56,168)$.

Size 9: $C(9,27) \times C(72,216)$.

Size 10: $C(30,10) \times C(270,90)$.

Sizes 5-10 are expanded below:

Size 5: $3003 \times 4.1918445058055 \times 10^{13}$
 Size 6: $18564 \times 6.7313297450658 \times 10^{22}$
 Size 7: $116280 \times 5.09429646650456 \times 10^{32}$
 Size 8: $735471 \times 1.79936382229943 \times 10^{44}$
 Size 9: $4686825 \times 2.94832030136639 \times 10^{57}$
 Size 10: $30045015 \times 2.232054795003 \times 10^{72}$

For the size 2 square, there are combinations of 2 out of 8 exterior and of 2 out of 8 interior holons required for each tile, yielding 28×28 combinations = 784 total possible tiles only for side 2, the only size in which exterior and interior holon numbers are the same. For size 3 and up, there are always more interior holons than exterior ones, because some of the squares which make them up are entirely inside the rep-tile. The side 3 square needs combinations of (3, 12) exterior and of (6, 24) interior holons, so $220 \times 134596 = 29611120$ total tiles. From here on I will use exponential notation to save space as the zeroes mount up.

Size 4 square: combinations of (4, 16) and (12, 48), equalizing $1820 \times 69668534468 = 126796732731760$ total tiles (c. 126 quadrillion...).

Size 5: $C(5, 20) \times C(20, 80)$.

Size 6: $C(6, 24) \times C(30, 120)$.

Size 7: $C(7, 28) \times C(42, 168)$.

Size 8: $C(8, 32) \times C(56, 224)$.

Size 9: $C(9, 36) \times C(72, 288)$.

Size 10: $C(10, 40) \times C(90, 360)$.

Sizes 5-10 are expanded below.:

Size 5: $15504 \times 3.53531614221217 \times 10^{17}$
 Size 6: $134596 \times 1.69745387607974 \times 10^{27}$
 Size 7: $1.18404 \times 7.57917178477062 \times 10^{43}$
 Size 8: $1.05183 \times 3.11639547214739 \times 10^{58}$
 Size 9: $9.414328 \times 1.17288418023316 \times 10^{74}$
 Size 10: $8.47660528 \times 4.02421012996903 \times 10^{92}$

Eventually it would be useful to list the calculated numbers for total tiles for squares and triangles for sizes from 2 up to a large size, as integer sequences.

It is eminently possible to program a computer to list all possible tiles of the rep-tile and size in question, and for this to be systematic, some reasonably logical holon numbering conventions will be vital to establish and maintain. As long as these conventions are maintained as we count sizes upwards from two, they can even be arbitrarily assigned, but as the finder I claim right of convention-setting, unless a much more logical approach is demonstrated! The *interior* anatomy of the numbering conventions is different for tiles of the triangle and of the square, but allows successful realisation of the same goal in each case, namely that *from each side of the rep-tile the exterior and interior holons are numbered in the same amounts and way.*

We begin with the rep-tile's bottom left corner and proceed counter-clockwise, counting and numbering all exterior holons; this process is the same for both triangles and squares. We then continue numbering the interior holons where we left off, as follows. For triangles, starting in the bottom left corner we count up the left side of the first interior line going up and right, down its right side, then move to the right if there are more holons and continue. When that side's interior holons are counted, we rotate counter-clockwise to the next side and continue in the same manner, until all interior holons are numbered. For squares, we count the interior holons only on the left side of each, not the right, before rotating and continuing.

By these means, at a glance we can distinguish any single holon's unique position using its number, which is simply the order in which it was discovered. We further simplify the view, in L-systems, by using F for exterior holons, and f for interior ones, G and g for exterior and interior spaces jumped, not drawn (differences not noticed by the L-systems parser); and in IFS, by following each holon's text string with a single semicolon for interior holons and a pair of semicolons for exterior ones (the semicolons being ignored by the IFS parser anyway).

I eventually hit upon a novel way of programming rep-tile tiles, which I have termed DNA (see Appendix). This would be most useful when we start using computer programming to output the codes for all tiles of a particular rep-tile and size. It works for both L-systems and IFS programming versions of the tiles.

For the DNA, ALL possible holons, in the numbering order above, are included in a master file for the desired rep-tile and size, and all are by default "turned off" by one of two methods. For L-systems, we program using G and g (which stand for non-drawn lines) instead of F and f; L-systems programming generally ignores differences between uppercase and lowercase. For IFS, we turn off each holon by placing a semicolon in front of the string for it. In each case, we copy the DNA file, name it uniquely for a particular tile, and then "turn on" only the holons we need for a particular tile simply by replacing G or g with F or f to turn on desired holons, in the case of L-systems; or removing the desired preceding semicolon for each holon we wish to turn on, in IFS.

Figure 6: The size 2 square, broken into its 4 smaller squares, showing the numbering convention for (single sided) exterior and (double sided) interior holon lines. The same for the size 2 triangle, which has only 12 holons compared to the square's 16.

The use of G or g (which tells the L-system parser to leave a blank space, instead of F or f which mean drawing a line segment) is crucial to the programming of many, though not all, of these tiles. Using G/g , one can move about the rep-tile from point to point without leaving a “trail” of line segments, until one is ready to begin drawing again using F/f . So the use of G/g here might be compared to the use of 0 in mathematics.

To calculate the necessary iteration number for a tile in an *L-system* render, one multiplies the size of the rep-tile by itself repeatedly until it surpasses the desired render size; each multiplication adds 1 to the iteration number (called the order number in FractInt). For example, if I am rendering a tile of a rep-tile of size 2 at 30000×30000 pixels, I need 2^{15} pixels on a side to achieve more than 30000 pixels, so, the iteration number is 15. For a rep-tile of size 3, it's $3^{10} = 10^{\text{th}}$ iteration, for size 4, $4^8 = 8^{\text{th}}$ iteration, and so on. These iteration numbers remain constant for any tile, so it is a good idea to write them down for future reference.

To calculate the necessary iteration number for a tile in an *IFS* render, one must simply experiment with several increasing maximum iterations of the tile at the desired render pixel size until the output render is sufficiently dense, i.e. usually filled in solid with pixels. Note the iteration count, and this is a constant for IFS renders at that render size.

Duped

Above I mentioned duplicate tiles, which exist in several different categories. One of these is when symmetries result in different holon sets producing the same tile, e.g. in mirror-image copies of each other, which can be considered as the same tile; these duplicates occur in all rep-tiles and sizes, and effectively cut the *total* calculated number of tiles for any rep-tile and size in half (Figure 7). Another duplicate category occurs in rep-tiles the side length of which is a positive integer exponent of another integer. In this case, for example, all of the tiles of a size 2 rep-tile will be found among those of its size 4, 8 and so on; all those of size 3 in size 9, 27 and so on. (The 2nd iteration of a size 2 tile, with $4^2 = 16$ holons, is the same as the 1st iteration of a size 4 one, &c; see Figure 8 below.) Also, in squares, if a tile has flip or mirror image symmetry about a vertical axis, its 180-degree rotated version will be a duplicate.

The duplication of smaller size tiles in larger sizes, as explained above, means that if we wish to refine one of the former, we can choose its 2nd iteration to work on, which is equivalent to its first iteration in a size which is the square of its original size. The larger tile has many more holons to play with and alter, indeed it has the square of the number of the smaller tile's holons. We can make smaller refinements by using these holons as each of them is the square root in size of the original's holons. If the 2nd iteration is still too coarse, we can go to a still higher iteration...

It is also possible to order the holons of a particular tile in a huge number of different ways, all leading to exactly the same visual result. For this reason, once a tile is “discovered”, no matter how its holons are arranged, it should be classified as discovered, its finder noted, and all other ways of ordering its holons excluded (unless some of these are otherwise significant, such as is the case with rep-tile-filling tiles, in which the fractal texture resulting from different holon arrangements is important). My own > 10000 tiles are in this “found” category, although they are a somewhat messy group, most not following the holon orders which I suggest in this article, many having been programmed before I standardised the production rules.

Although these tiles, as mathematical objects, are not able to be copyrighted in either “production rule text string” or graphic form, the finder of each can be noted. If finding them ever becomes popular, a race may develop to find the most tiles, aided perhaps by superior computing skills, and a webpage be set up to record finders and numbers. But when the field is literally infinite, no one can actually claim ALL of them! Anyone can find as many tiles as they desire; perhaps whole new subsets await discovery just in 2 dimensions, while 3 dimensional tiles of the cube are still untouched. My plea is for standardisation to be observed (either mine, documented here, or a better one if it is found), though, to avoid chaos in exploring these infinite sets.

Patterns

Given that there are a mix of unique and duplicate tiles for any rep-tile and size, one might expect to find patterns of unique ones and duplicates emerging from the total set of both. These patterns could follow a particular equation or number series, such as that of the primes in the natural numbers, assuming a standardised method of counting them, in which “zero”, or the starting point, and the direction of moving from one tile to another were agreed on. This possibility of patterns is a whole other area aside from the excitement and wonder of the tiles themselves: their order of occurrence.

Figure 7: Mirror-image versions of the same tile, which for all intents and purposes might be considered the same tile. The presence of both means that the *total* tile set for a particular size of rep-tile can be cut in half as we move towards a set of *unique* tiles.

Figure 8: Iterations 1 and 2 of a size 2 triangular tile; the 2nd image can also be considered, and in fact found, as iteration 1 of a size 4 tile. This means that all size 2 tiles can be removed from all sizes which are integer exponents of the same rep-tile, to close in on their sets of unique tiles: in this case, sizes 4, 8, 16 and so on.

One might expect that the set of holons for tile a , but merely rotated around the rep-tile to the next side, would produce the same tile a but itself rotated. Not so! This idea of “rotation sets” was a great surprise to me as I discovered that rotating a tile does not usually produce the same tile in a different position, accounting for some of the huge number of different tiles which any rep-tile contains. I made more discoveries upon the further understanding that interior holons are all double-sided lines, and that these two sides can be separated and only one of them used in a particular tile. Also, that holons need not even be connected to each other at all in the tile; each realisation furthered the possibilities.

It is easy to remove the first type of duplicates, arising from symmetries, from the calculated total, cutting it in half; the second type, based on exponents, are also a known quantity and can be removed by simple subtraction. The duplicates arising from rotation symmetries are groups which currently must be observed or calculated based on observations in the smaller rep-tiles’ sizes, and then also subtracted. The remaining set of tiles, with all duplicates “sieved out”, will then be all of the UNIQUE tiles for that rep-tile and size.

What is clear is that the total number of tiles for a particular rep-tile, all side lengths included, is a countable infinity. If the rep-tile sizes can be increased simply by starting at size 2 and incrementing by 1 towards infinity, and if there were only 1 tile per size, this already would be infinity. But as calculations above show, the number of tiles per size increases very rapidly as the size increases anyway.

Subsets

Rep-tile tiles can be divided into subsets in various ways, just like whole numbers can. One way is based on the amount of connectedness of their holons. This can range from: full or simple connection (the tile is all one piece); to weak or “wasp-waisted” connection, wherein an infinite number of

pieces of the tile are joined at points; to “non-zero size dusts”, in which the tile’s pieces are disconnected but are also of various sizes; to space-filling “pure dusts”, made of an infinite number of single holons at the limit. Outside these there are some atypical tiles made of less than an infinite number of pieces at the limit. It is currently unknown if there are an infinite number of these, or less.

There is also at least one ubiquitous tile of the square, i.e. found in every size: a right isosceles triangle. And there is at least one of the triangle, found in all sizes: a solid fill made by using all-parallel holons, so, not really a tile. It is not currently known if there are more ubiquitous tiles.

Another subset is formed by making all exterior segments “direction-reversed”, i.e. facing outwards instead of inwards. This is done simply by replacing all exterior Fs with G|F|G, FFs with GG|FF|GG., etc. It works for all tiles with 1) x-axis reflectional symmetry and with 2) all exterior Fs on 1 straight line; and for all all-parallel-holon tiles meeting condition 2. The resulting tile has $\frac{1}{2}$ the area of the original tile it was derived from: $\frac{1}{8}$ of the square for non-all-parallel tiles, $\frac{1}{4}$ of the square for all-parallel-holon types, etc. This type of tile subset has the designation letter r for reflection.

A couple more points are worth making. One is that rep-tiles of sizes which are not integer roots of integers, e.g. 2, 3, 5, 6, 7, 10, etc., offer higher proportions of unique to duplicate tiles; and the lower numbers in this set are the more “basic” or “fundamental” sizes, with tile sets which offer more “tweaking” in their duplicate versions in sizes which are integer exponents of them, such as 4, 8, 9, 16, 25, etc. Also, single-sided interior holons tend to show up at the limit as straight lines inside the tile, parallel to one or more sides, and also they reveal most clearly the size of the parent rep-tile, these lines often being of the same length as the original holons making up the size of the rep-tile. If one wants to obscure this size, for aesthetic reasons, one should avoid single-sided interior holons, although other considerations, such as the tile’s design, may force or encourage using them.

Figure 9: This odd example (1st and 2nd images) tiles the square in pieces which each comprise one whole tile and two half tiles, as the set of 9 copies below it shows. The latter image has the original tile at the upper left third of the whole. It is currently unknown if this is the only such tile. Next, 3 pairs of tiles showing the effect of reversing the exterior holons to face outwards; in each pair, 1st is the regular tile, followed by its altered version, in which something of the original can still be seen.

Figure 10: Examples of a simply or fully connected square tile; another wasp-waist-joined one (of a triangle); a nicely organic square tile of an infinite number of non-zero-size pieces; and an atypical square tile of only 2 pieces.

More subsets

Tiles of the square fall into another pair of categories, those occupying the “standard” $\frac{1}{4}$ of its area and those occupying $\frac{1}{2}$. Tiles of this latter type are formed only by using holons which are all parallel to one another. The nice thing about this type of tile is that by rendering it in a particular solid colour, you automatically render its opposite at the same time, because it takes up half of the square. Using all-parallel holons in the triangle, however, results always in space-filling sets, the above-named ubiquitous “tile” of the triangle.

Technically, tiles with all-parallel holons DO fill their rep-tile in the “standard” numbers of 4 per square and 3 per triangle. But at the limit they also cross themselves at an infinite number of points, i.e. everywhere; so they can also be considered as needing only 2 pieces for the square and 1 for the triangle, without any of these infinite number of crossings, with zero crossings, in fact.

Tiles can also be grouped in “rotation sets”. These are sets of the same tile but in all rotations about its rep-tile’s centre, one tile per side: three in a set for triangle tiles, four for square tiles.

Any tile can be given a range of colours, or a fractal texture, by colouring or shading its component holons by various rules. This may be a superfluous addition to many tiles, the edges of which are their main attraction and which need no further embellishment. But it is available if desired. One category of tile for which such colouring or texturing might be a great advantage is the ones which fill space in one piece, as they otherwise result in a solid triangle or square. Another is those which are more regularly geometric, with less interesting edges, and which could thus benefit from a bit of “livening up”. There are an infinite number of ways of altering the colouring or texture, simply by rendering any such tile’s holons in any of the vastly many different orders available, changing the colour set, or using a different size of the same space-filling render.

With other people taking up where I have left off, further exploring the infinite worlds of these tiles, it may well be that some entirely new subsets and other features emerge, having waited only for discovery. Eventually, 3d offers the greatest finds, tiles of the cube being as of yet a blank slate, entirely not yet drawn on. But 2d is also infinite in scope, so new types of finds there as well are certainly quite within the realms of the possible. Are there more tiles of the square with *diagonal* flip symmetry? Is there a tile which is half of the famous TwinDragon Curve? Other types of symmetry?

Figure 11: Pairs of fractally textured versions of 3 different square tiles (the 3rd pair are also an example of a tile using some 1-sided interior holons); and 3 each of 2 different triangular tiles, all in greys; and some colour examples too, these all of size 2 space-filling squares. All the colour ones can be rendered in any set of colours, of course, and can also be combined with themselves in various ways to achieve rotational or flip symmetries of 180 or 90 degrees (such as the last pair demonstrate).

Animations

It is also possible to smoothly increment a set of tiles from one to another, resulting in an animation of tiles when the renders are seen in sequence. This is done by: 1) choosing the output size; 2) choosing a start and end tile for the animation; 3) programming iterations of incremental tiles at that output size, changing only 1 holon per increment, and rendering them all. We here consider the desired output size as not a higher iteration of a simple tile but as iteration 1 of a tile at that size, i.e. all 2nd-iteration tiles from size 2 are duplicated as 1st-iteration tiles of size 4, &c. The choosing of increments could be automated as well, speeding up the whole process greatly. This would be done by constraining the computer to stick to properly formed tiles according to the production rules (as opposed to any random set of holons) while making the set of increments.

Aesthetic spectra

With an infinite number of tiles from which to choose, would it be a surprise that some are more beautiful, more aesthetically pleasing, than others? One factor in this seems to be the amount of connectedness. It need not be full, or 100%, connectedness, but tiles which have their parts too scattered or disconnected have little to draw the eye. See some more of my favourite examples below. Another sliding-scale variable is how geometric versus how organic the tile is. With these criteria in mind, the main use for these tiles could be in “training” artificial intelligences, or AIs, to “learn” beauty by presenting them with one or more human-chosen spectra of most to least beautiful tiles in sets as large as one wants. AIs could also be trained to recognize tiles in general and these types specifically; to find and eliminate duplicate tiles from the total set for any rep-tile and size, reporting on any patterns of duplicate or unique tiles within the totals; and even to glean a standard set of holons to make any tile from seeing its finished form.

Other uses and thoughts

When one is speaking of literal countable infinities of objects, mappable to the natural numbers, obviously there is no end to finding them, even though they are a “smaller” infinity than the one comprising all shapes and colours which a square or triangle of particular size can contain, which is more akin to the set of real numbers. A unique “ID” tile for every person on earth who is now alive, or who has ever been alive? For every living creature on the planet, or for that matter for every atomic or subatomic particle in the universe, dark matter included? That’s still not infinity. Wherever repeating patterns are needed, either static or slowly changing like parquet deformations do, even in animations, these tiles are available in 1, 2, 3, 4, 6, 8 colours, any size. Just in the solid filled in tiles, not the fractally textured ones, there is often a fascinating interplay between figure and ground, positive and negative space. As mathematical objects, they also cannot be copyrighted, so whoever wishes to program, find and use them is free to do so. Mathematics is *supposed* to be beautiful.

The best way to explore these tiles might be a programme dedicated just to their finding and rendering. It would include the following features:

- choice of rep-tile and size in 2d or 3d;
- manual or automatic modes of tile production; the latter systematic and exhaustive in the chosen rep-tile and size, with output to L-systems, IFS or both *program*, not graphic render) versions of all found tiles;
- display of the rep-tile's complete "anatomy" of all its holons as L-systems lines (the default type for exploration of new tiles) , with selected holons for a particular tile indicated and unavailable holons shown as such, and in a separate panel, preview of rendering the fractal in a small size as the user adds holon by holon until a tile is made
- hyperlink to an online library of already found tiles with details of number, date, founder &c, updated in real-time;
- solid-filled or fractal-textured rendering options for L-systems and IFS renders;
- choices of render sizes, colours and file formats.

The door is open, and the sky's the limit. I have lived long enough with these countable but insistent infinities running through only my head, hardly shared or used. But a finite mind, not meant to handle endlessness, might become overwhelmed with so much, too much really, and need to pass it around for its own relief as well as for the potential benefit of others who can see its worth and make something of it. Whoever all of you are, take it and run.

Here is a link to online secure storage of all my L-systems and IFS files related to the tiles I have described. Anyone can download these files with the link, and explore the 10000 or so tiles in a fractal programme such as FractInt, and then begin adding more to them.

<https://ln2.sync.com/d1/ac22514e0/f6z9psgi-xwsyk7p9-vbj4v9dz-q5xq2768>

Figure 12: Just showing off now! The last 3 triangular ones are trying to emulate Cantor, von Koch and Sierpinski respectively. The final square one, after them, tiling in 4, includes a detail of itself which is a tile in 2.

Figure 13: I also had to include some of what I would consider my less or least aesthetically appealing finds among these tiles so far, to show that the aesthetic infinities range into both ends of the spectrum. The patterns in these ones are less obvious, or they are too scattered, geometric or boring.

References

- [1] Apophysis (Ronald Hordijk, Mark Townsend). Open source fractal flame editor , created in 2003.
<https://sourceforge.net/projects/apophysis/>
- [2] Fractint (Bert Tyler). Freeware fractal generator, created in 1988.
<https://www.fractint.org/>
- [3] Georgia Today. <http://georgiatoday.ge/>
- [4] Lindenmayer, A. “Mathematical models for cellular interaction in development.”, *Journal of Theoretical Biology*, 18:280–315, 1968.
https://en.wikipedia.org/wiki/Aristid_Lindenmayer

Appendix

Examples of L-system tile DNA, sizes 2, 3 and 4:

sq2DNA { ; Anthony Hanmer 17/11/2002

Angle 4 ; Key to all 2x2 Square tiles

Axiom c9f+c47f+c12f+c14f

f=G+g|g+G+G+g|g+G+G+g|g+G+G+g|g+G+GG

g=gg

}

sq3DNA { ; Anthony Hanmer 17/11/2002

Angle 4 ; Key to all 3x3 Square tiles

Axiom c9f+c47f+c12f+c14f

f=G+ggg|ggg+GG+G+ggg|ggg+GG+G+ggg|ggg+GG+G+ggg|ggg+GG+GGG

g=ggg

}

sq4DNA { ; Anthony Hanmer 17/11/2002

Angle 4 ; Key to all 4x4 Square tiles

Axiom c9f+c47f+c12f+c14f

f=G+gggg|gggg+G+gg|gg+GG+G+gggg|gggg+G+gg|gg+GG+G+gggg|gggg+G+gg|gg+GG+G+gggg|gggg+G+gg|gg+GG+GGGG

g=gggg

}

Examples of IFS tile DNA, sizes 2 and 3:

sq2DNA {; 2x2 Square Tile DNA: Anthony Hanmer 20/11/2003,

; Designed with BRAZIL

; This IFS contains all segments of EVERY possible 2x2 Square tile,

; in the same order as the DNA for L-systems versions. (Here, exterior

; segments are marked with a ;; double semicolon.)

; Copy it, and activate the 4 desired lines by removing the

; preceding ; semicolon(s); and they will form the required 2x2 tile IFS.

;

;;0.5 0 0 0.5 -3 -0.5 0.0625;;

;0 -0.5 0.5 0 -0.5 2 0.0625

;0 0.5 -0.5 0 0.5 2 0.625

;;0.5 0 0 0.5 3 -0.5 0.0625;;

;;0 -0.5 0.5 0 5.5 2 0.0625;;

;-0.5 0 0 -0.5 3 4.5 0.625

;0.5 0 0 0.5 3 5.5 0.625

;;0 -0.5 0.5 0 5.5 8 0.625;;

;;-0.5 0 0 -0.5 3 10.5 0.625;;

;0 0.5 -0.5 0 0.5 8 0.625

;0 -0.5 0.5 0 -0.5 8 0.625

;;-0.5 0 0 -0.5 -3 10.5 0.625;;

;;0 0.5 -0.5 0 -5.5 8 0.625;;

;0.5 0 0 0.5 -3 5.5 0.625

;-0.5 0 0 -0.5 -3 4.5 0.0625

;;0 0.5 -0.5 0 -5.5 2 0.0625;;

}

```

sq3DNA {; 3x3 Square Tile DNA:  Anthony Hanmer 1/12/2002,
; Designed with BRAZIL
; This IFS contains all segments of EVERY possible 3x3 Square tile,
; in the same order as the DNA for L-systems versions.  (Here, exterior
; segments are marked with a ;; double semicolon.
; Copy it, and activate the 9 desired lines by removing the
; preceding ; semicolon(s); and they will form the required 3x3 tile IFS.
;
;;0.333 0 0 0.333 -4.002 -0.667 0.1111111;;
;0 -0.333 0.333 0 -2.337 0.998 0.1111111
;0 -0.333 0.333 0 -2.337 4.994 0.1111111
;0 -0.333 0.333 0 -2.337 8.99 0.1111111
;0 0.333 -0.333 0 -1.671 8.99 0.1111111
;0 0.333 -0.333 0 -1.671 4.994 0.1111111
;0 0.333 -0.333 0 -1.671 0.998 0.1111111
;;0.333 0 0 0.333 -0.006 -0.667 0.1111111;;
;;0.333 0 0 0.333 3.99 -0.667 0.1111111;;
;;0 -0.333 0.333 0 5.655 0.998 0.1111111;;
;-0.333 0 0 -0.333 3.99 2.663 0.1111111
;-0.333 0 0 -0.333 -0.006 2.663 0.1111111
;-0.333 0 0 -0.333 -4.002 2.663 0.1111111
;0.333 0 0 0.333 -4.002 3.329 0.1111111
;0.333 0 0 0.333 -0.006 3.329 0.1111111
;0.333 0 0 0.333 3.99 3.329 0.1111111
;;0 -0.333 0.333 0 5.655 4.994 0.1111111;;
;;0 -0.333 0.333 0 5.655 8.99 0.1111111;;
;;-0.333 0 0 -0.333 3.99 10.655 0.1111111;;
;0 0.333 -0.333 0 2.325 8.99 0.1111111
;0 0.333 -0.333 0 2.325 4.994 0.1111111
;0 0.333 -0.333 0 2.325 0.998 0.1111111
;0 -0.333 0.333 0 1.659 0.998 0.1111111
;0 -0.333 0.333 0 1.659 4.994 0.1111111
;0 -0.333 0.333 0 1.659 8.99 0.1111111
;;-0.333 0 0 -0.333 -0.006 10.655 0.1111111;;
;;-0.333 0 0 -0.333 -4.002 10.655 0.1111111;;
;;0 0.333 -0.333 0 -5.667 8.99 0.1111111;;
;0.333 0 0 0.333 -4.002 7.325 0.1111111
;0.333 0 0 0.333 -0.006 7.325 0.1111111
;0.333 0 0 0.333 3.99 7.325 0.1111111
;-0.333 0 0 -0.333 3.99 6.659 0.1111111
;-0.333 0 0 -0.333 -0.006 6.659 0.1111111
;-0.333 0 0 -0.333 -4.002 6.659 0.1111111
;;0 0.333 -0.333 0 -5.667 4.994 0.1111111;;
;;0 0.333 -0.333 0 -5.667 0.998 0.1111111;;
}

```