Conway Still Life Drawing

H. A. Verrill

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

Abstract

This article describes a method of producing high density still life patterns, with overall appearance of simple shapes, such as letters. These patterns are stable under the iteration of Conway's game of life rules. Some further artistic processing of these patterns is also described.

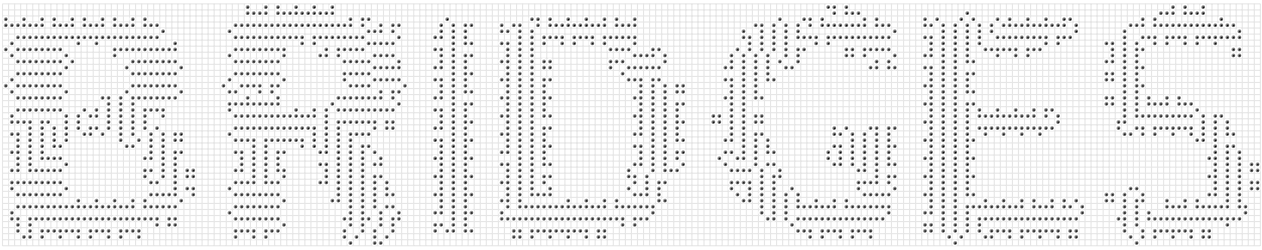

Figure 1: *BRIDGES in still life form.*

Conway's Game of Life

Conway's game of life is a famous cellular automata first investigated by John Conway in 1970 [7, 2]. There is a huge variety of beautiful cellular automata, with extensive investigations by Wolfram [13, 8] on two- and three- dimensional cellular automata. Complex behaviour can be observed from a simple rule set; virtual computers can be built with Conway's game [9]. The patterns have been used for their visual interest since discovery, for example computer screen savers.

Conway's game of life is played on a grid of squares as follows. Cells can be either alive or dead, and all 8 adjacent squares are neighbours of a cell. We start with an initial configuration of live and dead cells at time 0. Time is incremented discretely in integer steps. If a cell is alive at time k and has 2 or 3 live neighbours, it remains alive at time $k+1$. Otherwise it dies. A dead cell will become alive at time $k+1$ only when it has exactly 3 live neighbours at time k.

This article is limited to artistic applications, and does not address the mathematical theory. Other art based on Conway's game of life as still life includes work by Bosch and Olivieri [3]. They use a modular approach, finding a beautiful set of highly symmetric square still life tiles, with different densities. These are used to reproduce an image, such as a photograph, by replacing a square in the image with a still life tile of the same average density. The work in this article differs in that the artwork produced here is non-modular, produced by an iterative algorithm, with live cells having approximately 50% density throughout the shapes created. The image can not be divided up into tiles. Many other life inspired and related artworks can be found in [1].

In the first part of this article live cells are marked with a black dot and dead cells with a white dot, all on a white background. For aesthetic reasons, grid lines are not shown in most figures. The image at the end of the paper shows a version with squares instead of circles. The last part of this article is about using textures to interpret the underlying structure of a life pattern.

Still Life

A configuration of cells in Conway's game of life is called a "still life" [2, p. 819] if it is unchanged on application of the rules. This means the configuration at time $k + 1$ is the same as that at time k.

Noam Elkies [6] proved that infinite still life configurations have a maximal density of one half, where density is the ratio of live cells to the total number of cells. In [4] the authors construct arbitrary sized squares filled with maximal density still life configurations, and show that the maximum possible density may by slightly larger than 50% in a finite region.

In this article, instead of filling squares with maximal density still life configurations, other simple shapes are filled with still life configurations with density close to 50%. Letters inspired by the Demaines' font collection [5] are shown in Figure 1. Other simple shape are shown in Figure 2. These figures are solutions to a puzzle. Each dot has a special relation with its neighbours, and in most cases the configuration is a united whole, each part being dependent on another. This concept is known as the still life being "strict": if any part is removed, the remaining piece is no longer a still life. Each figure is the solution of an optimization problem. The quantity to be minimized is the sum over all cells of the absolute value of the difference between the value of the cell, and the value which would be obtained after application of the automata rules to all cells. In a still life situation, the values are unchanged, so this sum is zero. As well as the final artistic results, the process of producing these, viewable in animation at [10] is a dynamic artwork.

Figure 2: *Shapes of still life, generated using the algorithm illustrated in Figure 4.*

Not Just Life or Death

This section introduces a modified game of life. Each cell has a **life value**, a number ℓ between 0 and 1. Just as in many computer games, your "life" has a value which can go up or down. At each step of the game, ℓ either changes by the **life increment**, δ, or remains the same. In the usual Conway game of life, $\delta = 1$.

We introduce a second variable, the **life threshold**. This is a number L, between 0 and 1. A cell is alive if $\ell \geq L$ and dead otherwise. In the usual Conway game of life $L = 1$.

The rules of the game are that if a live cell has 2 or 3 live neighbours, its life value becomes $\min(\ell + \delta, 1)$ at the next step, and if a dead cell has 3 live neighbours the life value becomes $\min(\ell + \delta, 1)$. Otherwise the life value becomes $\max(\ell - \delta, 0)$.

Conway's game of life is obtained with $L = \delta = 1$. If δ is very small, the cells become more alive very slowly, whereas the game is more dynamic with larger δ. If L is very small, say 0.2 it is easier for cells

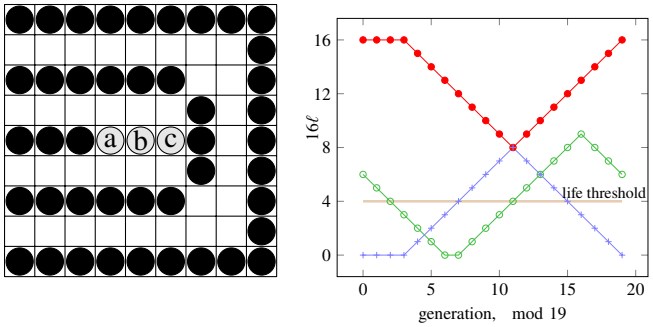

Figure 3: *A period* 19 *oscillating configuration, with* $(L, \delta) = (\frac{1}{4}, \frac{1}{16})$. *Cells labeled* a, b, c *on the left have life values* ℓ *given by the red, green, blue, graphs respectfully on the right.*

to become alive, so we get high density (approx. 50%) still life patterns evolving. When we revert to the original case of $L = \delta = 1$, if we start with a sparse randomly distributed collection of live cells, we tend eventually to have a sparse still life evolve.

Lemma. For any still life configuration, all cells have life value either 0 or 1.

Proof. The rules of the game replace the value ℓ of a cell with $\ell + \delta$ or $\ell - \delta$, unless $\ell = 0$ or 1. Therefore stability from one generation to the next is only possible if $\ell = 0$ or 1 for all cells. □

Theorem. Still life patterns for any given value of L and δ are also still life patterns for $L = \delta = 1$.

Proof. In a still life configuration, a cell is live if and only if it has two or three live neighbours. By the lemma, that is neighbours with value 1, which is the same for any value of L and δ. Similarly for dead cells. Therefore the still life patterns are independent of L and δ. □

Periodic situations can evolve, as shown in Figure 3. Black circles have life value 1; white cells have value 0. Gray circles have values shown in the graph. The brown line indicates the life threshold. When one graph drops below this line, or is no longer below this line, another graph changes gradient. Different values of the parameters give rise to different periodic cases. Every game on a finite board ends up either as a still life or a periodic configuration. Since a periodic configuration depends on the parameters, we can change L and δ to destabilise it. Periodic configurations can persist even when the parameters are perturbed. However, experimentally, by varying the parameters sufficiently a high density still life results.

Generating Still Life Drawings

To obtain a still life pattern filling a particular shape we use the shape as a mask. In the mask area we choose one set of values for L and δ e.g., $0.04 \leq L \leq 0.2$, $\delta = 0.001$, and outside the mask area, we choose another set, e.g., $L = 0.9$, $\delta = 0.01$. Away from the boundary between the masked and unmasked areas, the configuration of live and dead cells tends to stabilise fairly quickly if L is small for the mask area, and large for the non-mask area.To avoid periodicity, in the masked area we multiply L by a random number between 0.0 and 1.0 at each application. This tends to discourage periodic patterns, which depend on L and δ.

The initial configuration provides randomly distributed live cells in the mask area, with all cells outside the mask area dead. As long as the life value L in the non-mask area is not too small, e.g., over 0.8, and the δ value outside the mask is very small, e.g., 10^{-2} then the area outside the mask remains dead, except possibly on the boundary of the mask, because with the great relative difference in L inside and outside the mask, it may take thousands of generations longer for the cells in the unmasked area to reach the life value, but meanwhile, the cells in the masked area reach a stable state more quickly. Even if some cells outside the

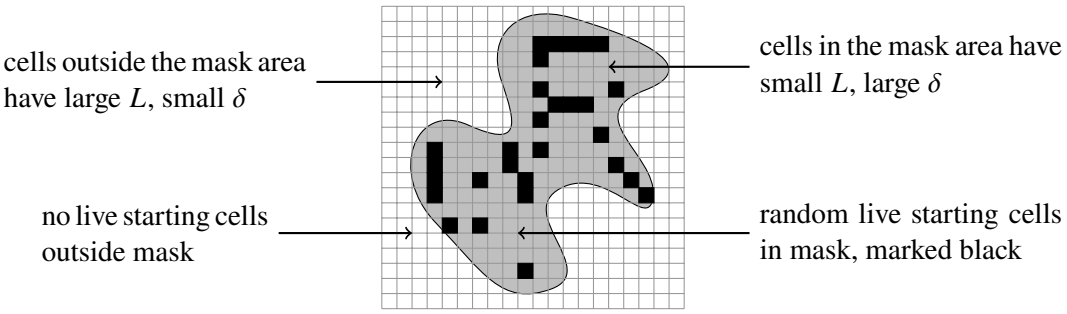

Figure 4: *Sample initial state of the algorithm for drawing still life.*

mask become live, it takes exponentially longer for each step further from the mask area for a cell to become live, so it's very unlikely for cells far from the mask to become live. A small δ value is taken rather than zero, to allow the possibility of a still life with a small number of adjacent live cells from outside the mask area. Changing L in the unmasked area even very slightly can have a big effect on the stability of configurations at the boundary, and so can be changed to disrupt and destroy oscillating patterns on the boundary. It is better to change the values inside the mask area, since changing those outside will generally increase the area populated by live cells. In order for the pattern to stabilise to a still life situation more efficiently, at each step a randomly selected set of cells is updated to have a new life value depending on their neighbours. Thus the cells are gradually adjusted to have a still life relationship to each other.

To summarize, the algorithm, also illustrated in Figure 4, is as follows:

1. Assign random values, distributed uniformly between 0.0 and 1.0 to cells within a shaped mask, leaving all cells outside the mask zero.
2. Choose L and δ for the masked and unmasked areas.
3. Compute the number of live neighbours of all cells.
4. Make a random choice of cells and update their life values simultaneously.
5. Repeat steps 2–4 until a still life configuration is achieved.

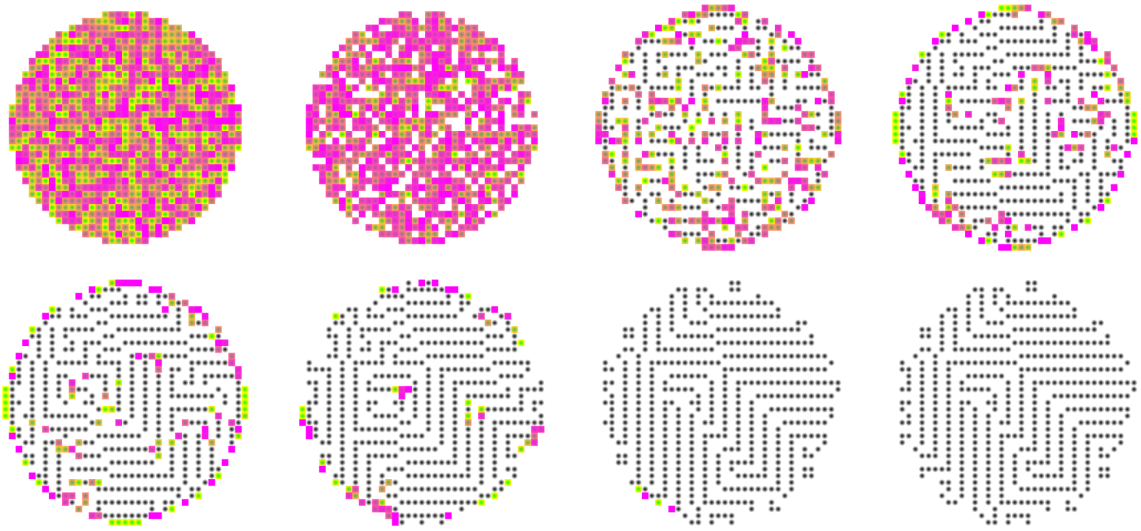

Figure 5: *An example of some of the steps of the algorithm on the way to a high density circular still life.*

Some of the steps during application of the algorithm to a circle are shown in Figure 5. In this figure, the black dots have life value 1, the dots are not drawn if the value is 0. Cells with life value strictly between 0 and 1 are magenta or green. Magenta dots have life value less than half, and green dots have life value greater than half. Thus the figure will represent a still life when it becomes monochrome. If the grid is too small, the result may not bear so much resemblance to the mask, as in Figure 7, right. In Figure 5, the images in the first row are obtained after a few seconds. The last row takes longer to work through, taking several minutes and adjustment of the parameters before the final still life is obtained. Some figures take much longer to stabilise. In Figure 6, the letters A on 40×40 grids were produced after approximately 2×10^7, 10^8, 3×10^7, 10^8, 3×10^9, operations on cells respectively, meaning this was the number of times a cell was considered and may have had its value changed. This took approximately 3, 20, 4, 25 minutes and 3 hours respectively. For the fourth case, the parameters were $\delta = 0.0001888$, $L = 0.11$ inside the masked area, and $\delta = 0.0137$, $L = .900$ outside the masked area, for the duration of the run, with no change. For the first case, the parameters were these until close to the end, when a change to $0.1513, 0.5956, 0.164, 0.969$ was made. For the other cases, more changes in parameters were made.

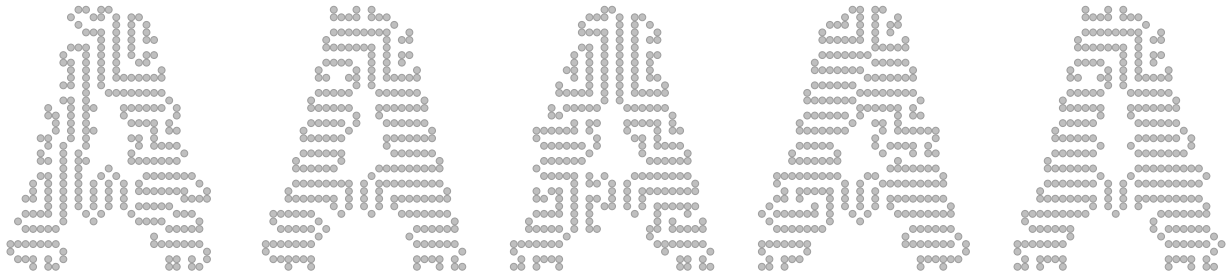

Figure 6: *Different still life "A"s produced by the algorithm.*

This is a random, rather than deterministic algorithm. The cellular automata itself generates the solution. Each time the program is run, it will give completely different outcomes, as in Figure 6. The grid size also affects the final result. In Figure 7 grids of different sizes are used. For the 100×100 grid case, the outline of the shape is very clear, and the inner structure becomes more interesting, with the eye finding lines where the pattern of dots change. For the 10×10 grid, the resulting pattern is very simple and bears less resemblance to the original letter B used as the mask.

Figure 7: *Results for high (*100×100*), medium (*30×30*), and low (*10×10*) resolution grids.*

As suggested by a referee, this algorithm can be interpreted as an optimization problem in the framework of simulated annealing. The quantity to be minimized is the number of "indeterminate" cells, i.e., those which would change on the next step of the game of life. The "temperature" can be given in terms of the values of L and δ, with higher values giving a higher temperature. Although the algorithm can terminate with no variation in these values, there is a tendency to get stuck at local minimum configurations. I have changed the algorithm to temporarily increase the "temperature" for approximately 0.025 seconds when the number of indeterminate cells falls below 3% of the total number of cells. This is done at most at 10 second intervals, to allow "cooling" between the temperature increase. This has vastly improved the efficiency of the algorithm, which now terminates in far less time, though I do not have a proof that the algorithm always terminates.

Hanging Textures on Still Life

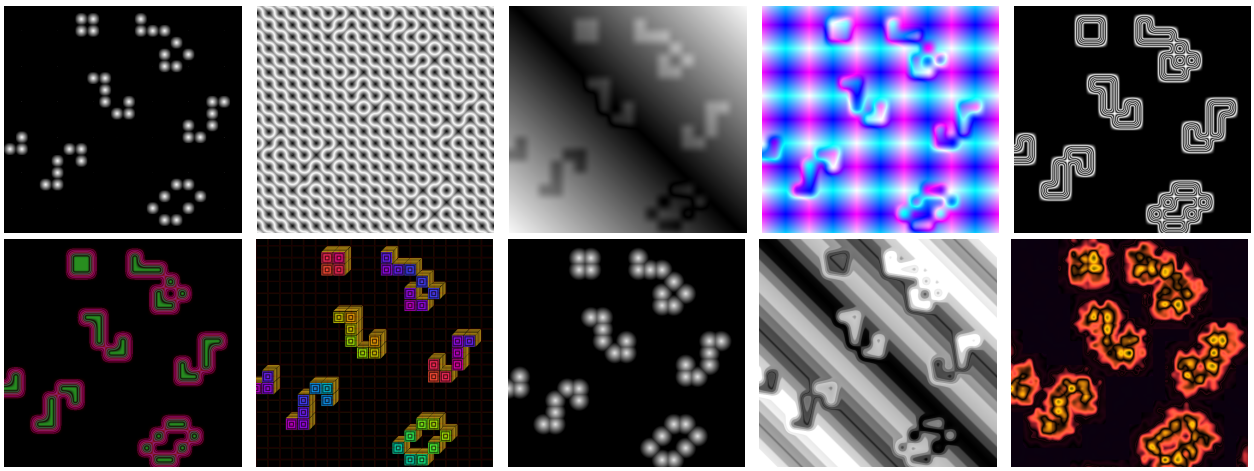

Figure 8: *These are all derived from the same configuration of low density Conway still life configuration.*

So far, our images have been based on taking an array of numbers representing life values, and filling the corresponding cell with a corresponding colour. However, the value of the cell in the array could be used to determine any number of possible parameters of an image in the corresponding cell in the drawing. If δ is very small, we have the visual appearance of a continuous value for life, and we could illustrate the value of life by the size of a circle in the cell. Or we could use the life value to determine the angle of a tile in a Truchet tiling or the height of a button. Examples are shown in Figure 9. We can also let tiles overlap, and use values of adjacent cells to determine the image. This is inspired by the method of Perlin noise [12], where values at the corners of a square determine the values in the interior by a smooth interpolation process. In this way, cells can gradually merge together, producing a more continuous image than the regular life pictures. Examples of this are shown in Figures 9 and 8. In the bottom right image of these Figures, Perlin noise across the image is used to add some randomness to the radius of the circles. Although the process used to determine the images could be applied to any array of numbers, these results have a pattern characterised by their origin in life, and so the resulting images reflect this. Because cells die with overcrowding, the overall density is not too high for individual cells to be made out in the images. Whereas in the original algorithm, the goal was a still life image, the dynamics of oscillating life patterns produce a beautiful motion. Moving forms can be seen by using the programs at [11].

Putting the textures of this section together with the still life algorithm of the previous section, we can now reinterpret the initial figure using any one of various possible textures, for example, as in Figure 10.

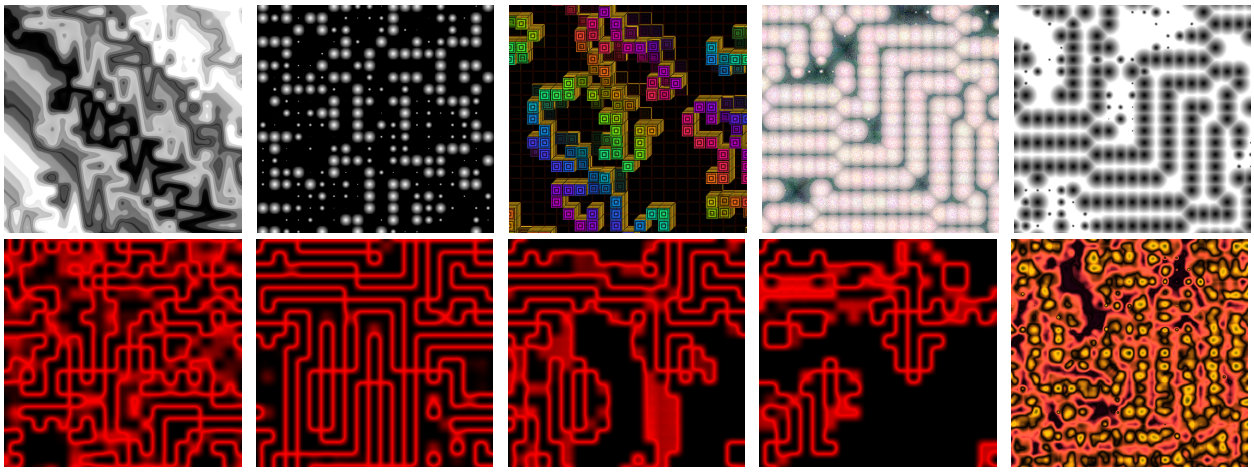

Figure 9: *These are stills from a dynamically changing variant of Conway's game of life, with small threshold value giving higher density configurations. The first four images in the second row give an idea of a dynamically changing image.*

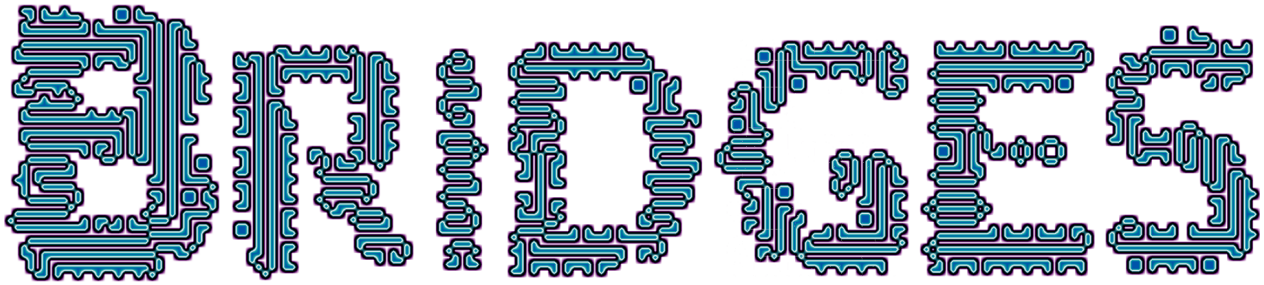

Figure 10: *"Bridges" still life with texture.*

The patterns in this section are textures programmed as shaders in WebGL, with the array of life values in JavaScript determining the parameters of the cells used for rendering the texture. The JavaScript/WebGL programs used to create the images in this paper are available at [10] and [11]. Each letter in Figure 1 was made from a single sans serif bold letter, placed on a 50×50 grid. In Figure 10 a 45×45 grid was used.

Conclusion

The first half of this article has clear objective of filling shapes with high density still life patterns, an obvious question to ask after reading about filling squares in [4]. The second part is less goal oriented, where the aim was simply to make pretty patterns based on the game of life. However, the second part was conceived before the first. The idea of using a continuous version of life was chosen in order to make smoother renderings than possible with only discrete values. After implementing these renderings, it was experimentally observed that high density patterns were obtained, and so the idea of changing parameters to obtain the shapes and letters was born. This illustrates the interplay of mathematics inspiring art inspiring algorithms.

Acknowledgments

I would like to thank the anonymous referees for their suggestions which have greatly improved both the article and the algorithm.

References

[1] A. Adamatzky and G. J. Martínez, "Designing beauty: the art of cellular automata," part of the *Emergence, Complexity and Computation series*, vol 20, Springer, 2016, ISBN 978-3-319-27269-6.

[2] E. R. Berlekamp, J. H. Conway, and R. K. Guy. *Winning ways for your mathematical plays,* Vol. 2, (first edition), Academic Press inc., 1982, 01-12-091102-7, Chapter 25.

[3] R. Bosch and J. Olivieri, "Game-of-Life Mosaics," *Bridges Conference Proceedings*, Seoul, Korea, Aug. 14–19, 2014, pp. 325–327. https://archive.bridgesmathart.org/2014/bridges2014-325.html.

[4] G. Chu and P. J. Stuckey, "A complete solution to the maximum density still life problem," *Artificial Intelligence,* Vol. 184/185, 2012, pp. 1–16.

[5] E. D. Demaine and M. L. Demaine, "Fun with Fonts: Algorithmic Typography," *Theoretical Computer Science,* volume 586, June 2015, pp. 111-–119.

[6] N. Elkies, "The still-Life density problem and its generalizations," *Voronoi's Impact on Modern Science, Book I,* P. Engel, H. Syta, eds.; Institute of Math., Kyiv 1998, Vol. 21 of Proc. Inst. Math. Nat. Acad. Sci. Ukraine, pp. 228–253.

[7] M. Gardner, "The fantastic combinations of John Conway's new solitaire game 'life'," *Mathematical Games, Scientific American,* Vol. 223 no. 4. 1970, pp. 120–123.

[8] N. H. Packard and S. Wolfram, "Two-dimensional cellular automata," *Journal of Statistical Physics,* vol. 38, no. 5-6, 1985, pp. 901–946.

[9] P. Rendell, "Turing Machine Universality of the Game of Life," part of the *Emergence, Complexity and Computation book series*, vol. 18, Springer 2016, ISBN : 978-3-319-19841-5.

[10] H. A. Verrill, April 2022, Conway Still Life Generator, https://www.mathamaze.co.uk/CA2022/.

[11] H. A. Verrill, GitHub, Feb. 2022, https://github.com/HelenaVerrill/ConwayGameOfLife.git.

[12] P. G. Vivo and J. Lowe, *The Book of Shaders,* https://thebookofshaders.com/11/, accessed 14.04.2022.

[13] S. Wolfram, *A New Kind of Science,* Wolfram Media, Inc., Champaign, IL, 2002, ISBN: 1-57955-008-8.

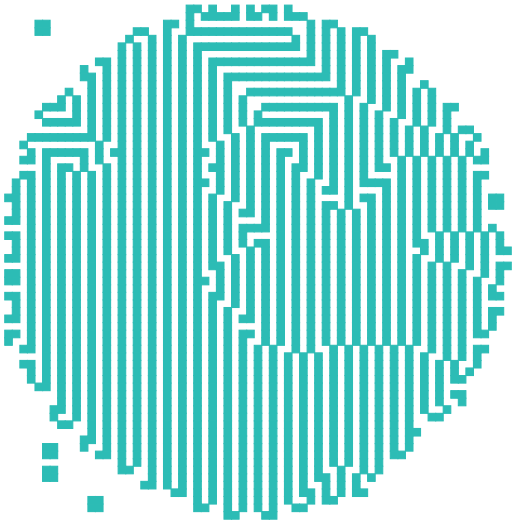

Figure 11: *Circular still life, with live cells marked by cyan squares.*