Column Me Maybe: Using Cellular Systems to Generate Novel Columnar Forms

Dustin Hines, Duncan Gans, David Anderson

17 December 2018

1 Introduction

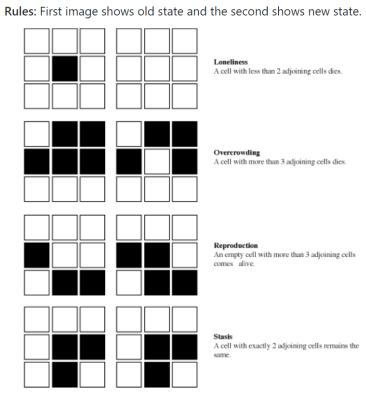

Figure 1: Rules for Conway's Game of life, a classic cellular automaton.

A long-standing debate in the sciences concerns whether machines can be defined that act in truly 'creative' ways. For instance, an early neuroscientist responded to Turing's claims regarding the potentials of AI by stating that "not until a machine can [act creatively] because of thoughts and emotions felt... could we agree that machine equals brain" (Jefferson 1949). As a result of such claims, AI researchers have long attempted to design systems that can act creatively and emotively. This research, generally termed 'Computational Creativity' has seen a great deal of development in the past twenty years as powerful computers and GPUs have become widely available (Colton and Wiggins 2012). For example, the popular game *Minecraft* employs procedural computer-controlled systems to design unique game-worlds on every playthrough (Persson 2011). In another case, a recurrent neural network named 'Benjamin' was used to generate the script to the short film *Sunspring* (Kortschak *et al.* 2016). These developments are not limited to the arts: in recent years, researchers have proposed and implemented computational creativity approaches into engineering tasks such as Computer Aided Design (CAD) and the design of efficient airplane wings (Krish 2011, Aage *et al.* 2017). As access to computational power and computational literacy continue to improve generative computational approaches may be expected to increasingly play a role in creative enterprises.

Cellular automata or "cellular systems" were initially developed conceptually by Von Neumann (1951) and Ulam (Beyer *1985*). In general, cellular automata consist of an n-dimensional array of 'cells' (e.g. pixels in 2D and voxels in 3D') with assigned neighborhoods. Every one of these cells is set to an initial state at time t=0. For every subsequent generation the state of every cell within the system is updated based on the states of the cells in its neighborhood and some predefined (typically simple) rules. While such systems are often simplistic in design, it is possible to generate complex behavior and patterns with the correct initial state. For instance, Conway's Game of Life (Fig. 1), a cellular automata developed in 1970, consists of binary cells and only four simple rules for updating cells. However, the game is able to generate incredibly complex dynamics. For instance, the game is computationally universal (Rendell 2000). The fact that cellular automata can produce such complexity has long fascinated computational scientists.

In recent years certain principles of cellular systems have been applied to creative endeavors. Notably, German architect Michael Hansmeyer has used generative approaches inspired by cellular division and L-systems to design columns, grottoes, and opera sets (Fig. 2, Hansmeyer 2012, Hansmeyer and Dillenburger 2017, Hansmeyer 2018). However, we were able to find no examples of the use of pure, fleshed out cellular systems for generative design in the literature.

We implemented two systems of cellular automata in Blender and test the efficacy of each for the design of columnar or sky-scraper like features. In brief, products the two systems consisted of a stacked series of states of each system and thus represented the behavior of the system over time. These objects were represented as a three dimensional array of binary voxels, where 'alive'

Figure 2: Column from 'Subdivided Columns', a generative architectural project by Michael Hansmeyer that was inspired by cellular division (Hansmeyer 2012)

voxels were rendered as 3D cubes. For example, the base of a column consisted of a two-dimensional layer of voxels representing the initial state of the system, while the next layer represented the subsequent state of the system and so on upwards. The first system, termed the 'deterministic system' operated in a purely deterministic, rule based manner where the state of a voxel and its neighbors at time n determines the state of that voxel at time $n+1$. In contrast, the second system operated in a partially stochastic manner, with the state of each voxel and its neighbors at time n determining the *probability* of the voxel surviving to time $n+1$.

Initial tests of each system revealed certain problems stemming from the chaotic nature of the generative process. Namely, the algorithms would often produce noise or rapidly expand or contract (that is, all voxels were alive or all voxels were dead). In cellular systems, this behavior is largely determined by the initial state of the system. In an attempt to generate more interesting columns we implemented a genetic algorithm (GA). This algorithm selected for initial states that generated columns that were denser towards central regions and less likely to expand in an explosive, uncontrolled manner.

To test the efficacy of these approaches for sculptural design, we rendered them in the popular 3D modeling software Blender and compared them to columns generated by randomly placing voxels in a column like manner. Results were mixed, partially due to the low resolution of the models. However, the stochastic approach did appear to be able to generate a few interesting models relative to simple random generation. This result suggests that simple cellular automata may be a useful paradigm to integrate fully into generative design.

In the second part of this paper, we describe in detail the design of our cellular automata and genetic algorithms. In the third section, we describe the methodology used to test the efficacy of each system for generating columnar sculptures. In the fourth section, we describe the results of our generative efforts. In the fifth and final sectin we discuss the implications of these results for future work.

2 Design of Cellular Systems and Genetic Algorithms

2.1 Translating Cellular Systems to Column Design

To begin with, we were faced with the challenge of how to translate the concepts of a cellular system to a columnar design. While we initially considered ideas involving the asymmetric division of cells (*a la* Hansmeyer 2012), we ultimately elected to follow a more simple approach. In our approach, columns consisted of layers of cubes or 'voxels'. Within each layer, each voxel corresponded to the state of a cell at a given iteration of a cellular system. In the bottom-most layer a voxel was filled if the cell it corresponded to was alive in the initial state of the system and empty otherwise. In the next layer up, voxels were filled if their corresponding cells were alive after one iteration of the system, while in the second layer up voxels were filled if their corresponding cells were alive after two iterations. In such a way the columns represented the behavior of a cellular system over time.

2.2 Implementation of the Deterministic Cellular system

We first attempted to build a purely deterministic cellular system for column generation. We represented our

system as a 3D array, where the nth layer from the bottom (i.e. the nth 2D subarray) represented the state of the system at time n.

The principle challenge in designing a deterministic system for column generation was determining what rules and initial conditions to use. To determine these, we took a genetic algorithms approach. We first initialized a population of cellular automata with 8 random rules for cellular survival (as each cell could have up to 8 neighbors). We then allowed each automaton to generate a column. These columns were evaluated by a fitness function that took into account height, centrality of the column (that is, how clustered the living cells were around the center vs. the edges of the rendered area), the existence of free-floating cells, and the amount of space filled by the column. The best columns were then selected and a new generation was created by crossing over their starting layers and rule sets with a constant crossover probability and randomly mutating the initial conditions and rule sets.

2.3 Implementation of the Probabilistic Cellular System

In the probabilistic approach we employed a somewhat different approach than in our initial deterministic attempts. For starters, we focused on engineering just one quarter-segment of the column. This was then rotated when rendering the final model, resulting in final renders that preserved radial symmetry by default.

This quarter-segment was represented as a square three dimensional matrix. Initially, the bottom-most layer (i.e. the first 2D submatrix of the 3D matrix) was initialized as a square such that some cells were labeled as 'alive' and other cells were labeled as 'dead'. Several such initializations were created and treated as a population of automatons.

We then allowed each individual automaton to run for one iteration according a a fixed set of rules. Our rules were as follows: if a cell had no neighbors, it died; otherwise, cells with 1, 2, 3, 4, 5, 6, 7, or 8 neighbors survived to the next generation with probabilities 0.1, 0.2, 0.3, 0.5, 0.8, 0.85, 0.9, or 1 respectively. The state of the automaton was then entered into the matrix representation as a new layer.

While we could have just allowed the system to run for an arbitrary number of iterations as described above, we decided to integrate ideas stemming from genetic algorithms into our generative process. Namely, after each iteration (that is, after each automaton has added a new layer) we crossed each new layer over with a random layer from another automaton. To do this, we compared each cell in the two automatons. If both were alive, the cell remained alive in the offspring. If dead in both, the cell remained dead in offspring. If alive in one but dead in the other, the cell remained alive 50% of the time.

We then ranked columns based on their fitness. Fitness was based on the total number of living cells in each layer (with higher scores being awarded to columns where layers were approximately 60 percent fill). This would ensure that the column wouldn't "die off" or expand to fill the entire screen. Fitness was also higher for columns with dead cells near the center and alive cells closer to the edges. This favors variation and pushes it away from being a circle. It essentially represents encouragement of slow explosion. These rules allowed us to select for high variation in columns without the columns expanding to fill the entire area or deleting entirely.

Having ranked our columns, we then selected the one with the best fitness and created a new population consisting solely of copies of that column. We then mutated these offspring by randomly switching column edge cells on or off in the topmost layer of the offspring at a constant probability. After all this, we allowed the system to run for another iteration and create a new layer. We repeated this process for a set number of iterations to generate a 3D column where each layer corresponded to one iteration and selection process of the system.

3 Testing Methodology

To test the efficacy of our algorithms we qualitatively assessed the ability of each cellular system to generate forms that were columnar while still being novel. To do this, we compared columns generated by each algorithm to columns generated by an algorithm that randomly placed cubes at each layer. We also assessed the ability of each algorithm to generate novel, unexpected forms while still maintaining a columnar shape.

When assessing the columnar shape of forms generated, we used several criteria. First, forms should be longer than they were wide. Given the nature of our algorithms this was more or less guaranteed. Second, forms should maintain a more-or-less even width along their length, neither expanding to the limits of the rendered area or dwindling away to a point. Third, forms should display a level of symmetry.

When assessing novelty, we looked for the presence of features unusual in traditional building and design practices, such as holes, arches, overhangs, and variation in surface texture. We also assessed the ability of each algorithm to generate unique forms based on different initial configurations. Additionally, forms were compared to columns generated by just randomly deciding whether or not cells in the system lived and died to determine the degree to which forms created by our systems

were the product of noise as opposed to the system itself.

While these measures are admittedly quite qualitative, they allow us to assess the efficacy of our purely cellular approach when applied to the generative design process.

4 Results

To test the efficacy of each cellular system for form generation we compared the forms generated by both systems to each other and to columns generated by just randomly selecting which cells survived from generation to generation. Results generated by the two cellular systems were mixed. Generally, the deterministic system did not generate columns that were markedly different than random columns. Often, the deterministic system produced forms that either dwindled after a few layers or rapidly exploded to fill the rendered area. By contrast, the stochastic system generated novel and unique designs which often displayed innovative characteristics and design ideas. While the design of the stochastic system has the advantage of preserving symmetry intrinsically, it generally had the advantage of producing more varied, pleasing and interesting designs.

4.1 Random Columns

We first generated columns by randomly setting cubes in each render layer. The columns generated by these processes were typically roughly in the form of a rectangular prism (corresponding to the bounds of the rendered area) and displayed a variable, pitted surface (Fig. 3). These models had some resemblance to architectural projects such as Habitat 67 (Safdie 1967). Generation of multiple columns revealed that random generation produced conceptually similar (though technically unique) columns on each run, suggesting that the random columns were good models of what a "noisy" column might look like and consequently a good point of reference for our two cellular systems. In general, while these columns are a useful point of reference they are not particularly interesting or novel from a design perspective.

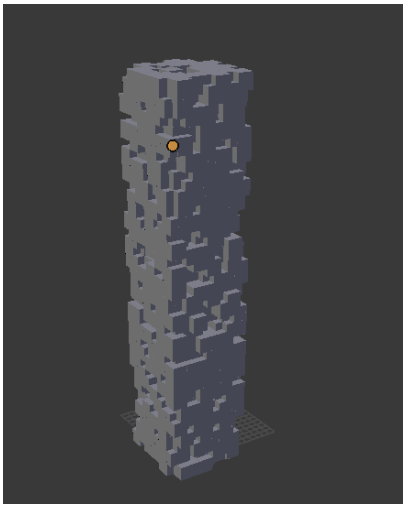

Figure 3: A randomly generated column

4.2 Deterministic Cellular System for Column Design

In general, the deterministic cellular system did not produce particularly interesting results. In many starting configurations the system rapidly dwindled and died, resulting in short, stunted columns. In most other cases, the system immediately expanded to the bounds of the rendered area in a noisy explosion, generating columns that generally resembled those generated by random processes (Fig. 4). Thus, in some sense, the genetic algorithm was successful at evolving cellular systems with broad characteristics: i.e. either systems that quickly died off or systems that rapidly expanded to fill the available space. However, we were unable to use the genetic algorithm to fine tune the deterministic cellular system to produce columns in between these extremes.

This is not to say that the deterministic approach was an absolute design failure however. Using this entirely generative, hands off process we were able to generate columnar forms. This served as a proof-of-concept going forward, as it informed us that cellular systems could be designed and translated into columnar structures using our general approach.

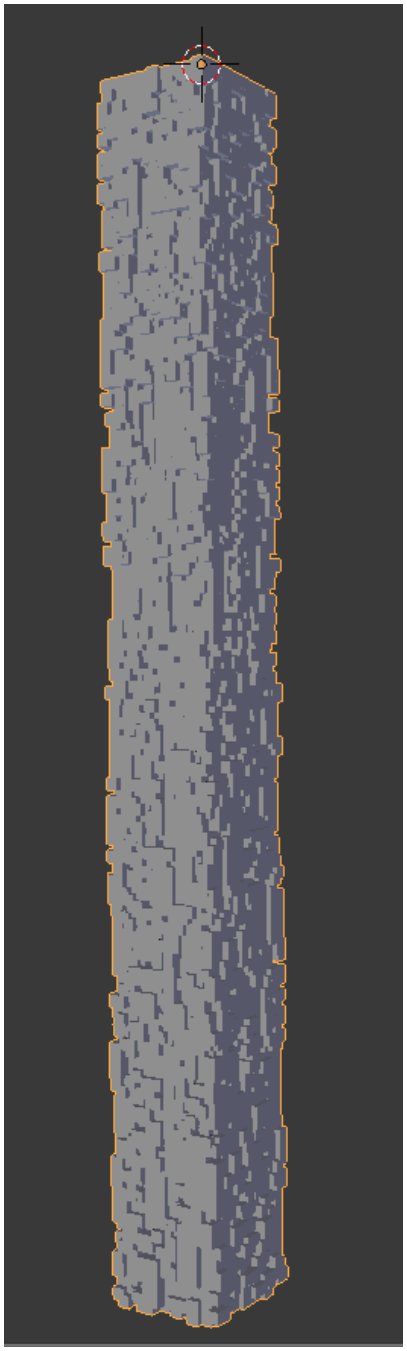

Figure 4: A column generated with the deterministic cellular system. While it successfully persists and is columnar in form, it does not display many novel characteristics when compared to columns generated randomly.

4.3 Stochastic Cellular System for Column Design

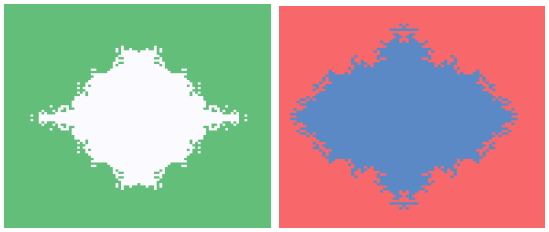

Figure 5: Cross sections from two columns generated with the stochastic cellular system. While conforming to a high level, round/prismoidal design, cross sections and columns displayed unique characteristics. In particular, edges appeared to resemble fractal boundaries observed in naturally occurring cellular growth.

In contrast to the deterministic cellular system, the stochastic cellular system produced several novel, unique and intriguing columnar forms (Fig. 6). Inital 2D renders revealed that these forms showed unique variation between cross-sections (Fig. 5). Initial results proved to be promising, and continued experimentation allowed us to generate many unique forms.

When we compared the forms generated by the stochastic system to randomly generated columns we saw little resemblance. While certain textural features were preserved between the two, columns generated by the stochastic cellular system remained constrained to the inner boundaries of the rendered area rather than exploding in a noisy fashion as was the case with randomly and deterministically generated columns.

Columns generated by the stochastic cellular system were generally columnar, measuring longer than they were wide and typically maintaining a generally even, though varying, width. A few exceptions to this were noted: for example, some columns split into two columns partially along their length, essentially resembling two towers merging into one.

Importantly, the stochastic cellular system was able to generate novel and unique designs. Figures 5-8 demonstrate some of the various forms created by the system. Forms displayed novel architectural elements such as hanging features, holes, and open interiors. In some situations the growth of the system got out of control and exceeded the bounds of the rendered area. However, in most cases columns were generated that were easy to render in Blender and could potentially be manufactured with modern 3D printing techniques.

While these results are qualitative, it appears that cellular systems can be reasonably directly adapted into 3D design approaches. Admittedly, in the case of the

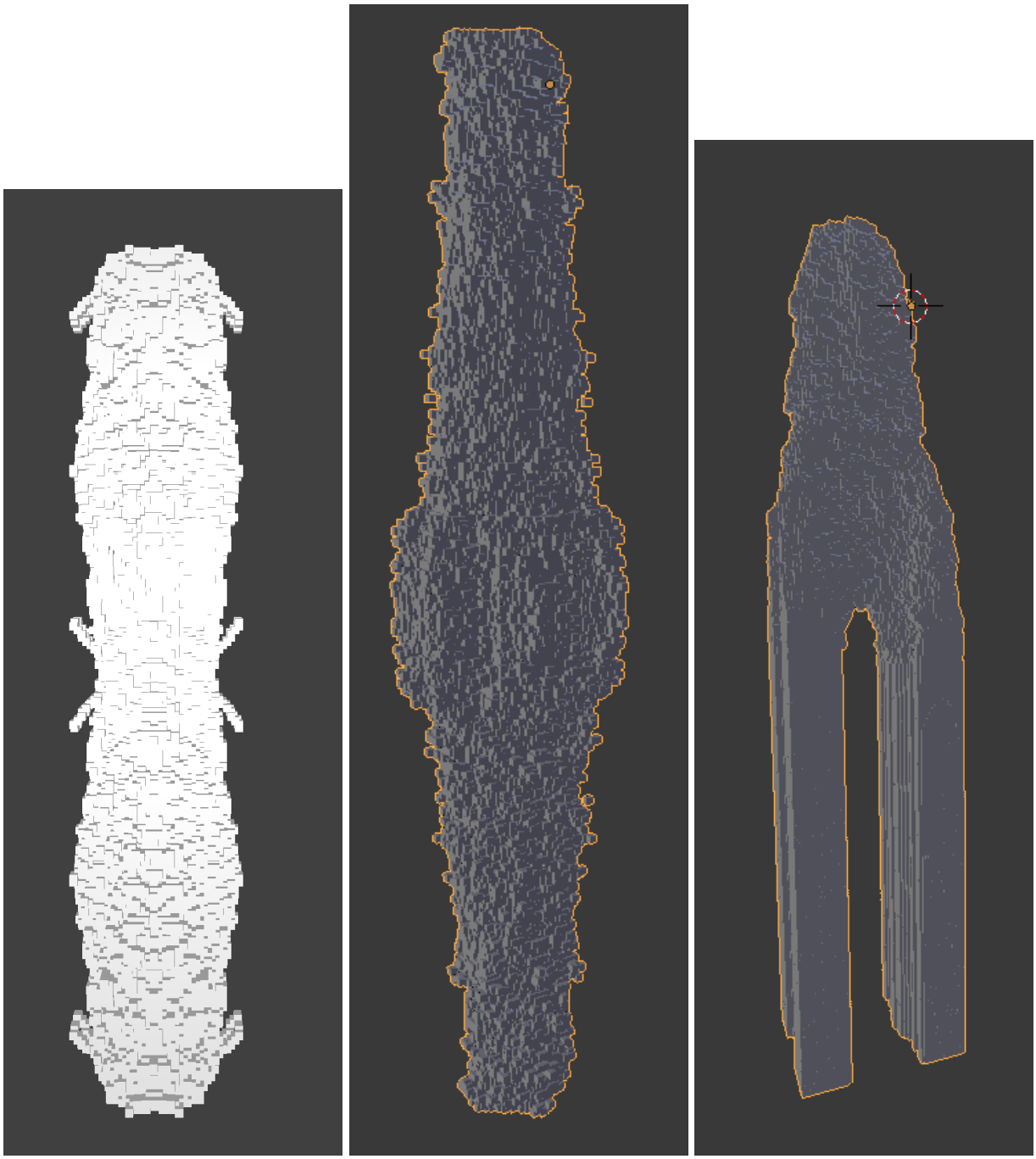

Figure 6: Three columns generated by the stochastic cellular system. In the column at left and the column in the center we see a generally columnal form and the generation of intriguing novel elements, such as the spikes in the leftmost column and the central bulge in the central column. In the column at right, we observed a split, where the column fully split into two parts before exceeding the boundaries of the rendered area, resulting in an arch-like form.

stochastic cellular system we tweaked our automata and renders in minor ways to produce more varied and pleasing designs. However, the brunt of the design work was purely generative.

5 Further Work and Conclusions

In this project we drew inspiration from cellular processes to build two cellular automata that create three dimensional columns in a generative manner. Additionally, we used a genetic algorithms based approach to select for cellular systems with initial configurations and rules that would generate columns with the desired qualities. While a purely deterministic cellular system did not generate particularly interesting designs, by introducing a level of stochastic uncertainty into the cellular system we were able to generate several novel and unique column designs. While processes inspired by cellular and biological processes have already been applied to architecture, this is the first direct adaptation of cellular systems to architectural and sculptural design that we are aware of.

Of course, this is only a first step. While the forms generated by our processes serve as a proof of concept, they also display several drawbacks. To begin with, our models are raster based and thus have a highly limited resolution. Models used in CADD are typically vector based, allowing them to be smoothed, scaled and printed effectively. Due to the raster based and low-resolution nature of our models they often appear to be rough and difficult to engineer in the real world. Future research should be directed towards adapting this cellular approach to vector-based formats or on improving the resolution of the approach to the extent where the raster-based format is not so much of an issue. For instance, one might imagine a program that generates parametric models of the raster models output by the cellular system.

It would also be fascinating to see if the objects generated by this process could be rendered in the real world. The level of detail and complexity present in any of the objects generated by our processes is far higher than that found in traditional building materials or styles. However, computer managed production approaches, such as 3D printing, could be employed to actually produce the models in the real world. We are currently experimenting with such processes; however, due to software constraints such as the inavailability of cheap, scriptable 3D modeling software that integrates well with 3D printers we have met little success thus far.

In general, computation creativity is coming increasingly to the forefront of creative endeavors. As the century progresses, computational designers will likely focus less on designing end products and more on designing processes that design these products for them. Here, we demonstrate that a creative approach inspired by cellular systems and genetic algorithms can be used to create columnar objects with potential uses in architecture and the arts. While the results are crude, they serve as a proof of concept for future research.

References

[1] Aage, N, E Andreassen, BS Lazarov, and O Sigmund. 2017. Giga-voxel computational morphogenesis for structural design. *Nature* 550: 84-86.

[2] Beyer, WA, PH Sellers, MS Waterman. 1985. Stanislaw M. Ulam's Contributions to theoretical theory. *Letters in Mathematical Physics* 10: 231-242.

[3] Hansmeyer, M. 2012. Building unimaginable shapes [Video]. Retrieved from: `https://www.ted.com/talks/michael_hansmeye_building_unimaginable_shapes?language=en`

[4] Hansmeyer, M, and B Dillenburger. 2017. Digital Grotesque II [3D Binderjet Prints]. Centre Pompidou, Paris, France.

[5] Hansmeyer, M. 2018. Zauberflote [3D Binderjet Prints]. La Monnaie, Brussels, Belgium.

[6] Jefferson, G. 1949. The mind of mechanical man. *British Medical Journal* 1.

[7] Kortschak, A, W Kortschak, A Swett, A Friedman (Producers), and O Sharp (Director). 2016. *Sunspring* [Motion Picture]. United States: End Cue.

[8] Krish, S. 2011. A practical generative design method. *Computer-Aided Design* 43: 88-100.

[9] von Neumann J. 1951. *The general and logical theory of automata* in L.A. Jeffress, ed., Cerebral Mechanisms in Behavior The Hixon Symposium, John Wiley Sons, New York, 1951, pp. 131.

[10] Rendell, J. 2000. This is a Turing Machine implemented in Conway's Game of Life. at http://rendell-attic.org/gol/tm.htm

[11] Safdie, M. Habitat 67 [Building]. Montreal, Quebec, Canada.

[12] Persson. *Minecraft.* 2011.