Self-organized 3D-printing Patterns Simulated by Cellular Automata

Yasusi Kanada

Dasyn.com
yasusi@kanadas.com

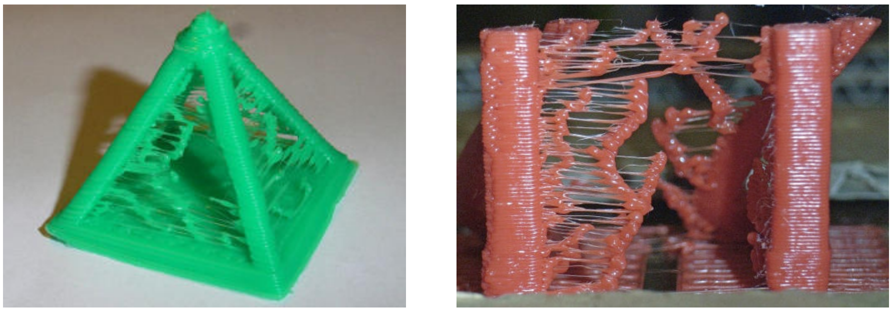
Abstract. 3D printers are usually used for printing objects designed by 3D CAD exactly, i.e., deterministically. However, 3D printing process contains stochastic self-organization process that generate emergent patterns. A method for generating fully self-organized patterns using a fused deposition modeling (FDM) 3D printer has been developed. Melted plastic filament is extruded *constantly* in this method; however, by using this method, various patterns, such as stripes, splitting and/or merging patterns, and meshes can be generated. A cellular-automata-based computational model that can simulate such patterns have also been developed.

1 Introduction

3D-printing technologies, or additive manufacturing (AM) technologies [Gib 10], usually aim reproducing objects deterministically designed by using 3D computer-aided design (CAD) tools. Object models designed by using CAD are horizontally sliced into thin "layers" by so-called "slicers", and a 3D printer prints the layers one by one. Especially, 3D printers for fused deposition modeling (FDM), such as those of Stratasys, Makerbot, or RepRap [Rep], shapes 3D objects by layering melted plastic filament extruded by a hot nozzle.

3D printing process contains self-organization process that generates emergent and fluctuated patterns but they have been ignored by the 3D-printing community. Printing processes contains bifurcations, and printing conditions and process including nozzle temperature, extrusion process, air motion, and so on, are fluctuated, so the generated patterns are partially self-organized and *naturally randomized* [Kan 14]. Although the printing process is usually controlled well so that the self-organization processes are suppressed and the fluctuation usually does not cause serious problems to shape 3D objects, stochastic patterns caused by fluctuation can still often be seen in printed objects as described below. However, self-organized patterns generated by 3D printers are regarded as noises and are mostly ignored in 3D printing communities and industries.

Self-organized stochastic patterns can be seen in printed objects such as shown in Fig. 1. Two types of stochastic patterns can be seen in this photo. First, thin *strings* exist between standing edges. Although filament extrusion stops when the head is to move without extrusion, it is difficult to stop it completely and an unintended string is often generated. Second, small *chunks* of plastic exist at the end or center of strings in Fig. 1. In contrast to strings, which are more uniform, the nozzle may create less uniform chunks. These patterns and other stochastic patterns, i.e., extrusion failure and sticking failure, are explained more in a previous study [Kan 14].



(a) Pyramid ($38 \times 38 \times 33\,mm^3$) (a) Between two objects ($38 \times 38 \times 28\,mm^3$)

Fig. 1. Printed objects with strings and chunks (ABS, by Rostock MAX)

The emergence of FDM printing processes can be stressed by designing a fully self-organizing printing process that simulates one-dimensional cellular automata (1D CA) and that generates artifacts including design itself as few as possible. This process was proposed by the previous study. It generates emergent and stochastic 2D patterns by helical print-head motion. Basic patterns generated by this printing method are stripes. However, stripes may sometimes spit or merge, waves may cross the stripes, and patterns may be meshes according to printing conditions. This study focuses on these types of patterns and shows a computational model based on 1D CA that can simulate two types of patterns and suggest processes of other types.

The rest of this chapter is organized as follows. Section 2 proposes a method for printing fully self-organized patterns and shows basic printed results. Section 3 proposes a CA-based computational method to simulate the basic patterns. Section 4 shows various types of patterns, an extension to the CA-based method, and compares patterns generated by the printing and simulation methods. Section 5 describes the differences between the printing and simulation results, and Section 6 concludes this study.

2 Method for Printing Fully Self-organized Patterns

To generate 1D CA-like patterns, a 1D space without edges is used ([Wol 84] [Kan 94] etc.) in the method proposed by the previous study [Kan 14]. The space occupied by the CA is topologically a circle. Therefore, to generate 1D patterns by an FDM printer, the print head can approximately draw circles in a clockwise or counterclockwise direction repeatedly and can extrude filament (Fig. 2). The speed of print-head motion and the speed of extrusion are constant to avoid artifacts, and a helix is used for the tool-path, i.e., the orbit of the print head, instead of layered circles because layer transitions create edges, or a type of artifacts, and spoil the pattern. This pseudo-layering method is quite different from conventional 3D printing method, which generates slices, i.e., complete layers, with variable head and extrusion speeds. In addition, because a head of a FDM 3D-printer can only moves linearly, a "circle" is approximated by a collection of line segments. Although the extrusion velocity is constant, if the printing condition is selected carefully, the printer can generate stochastic self-organized patterns.

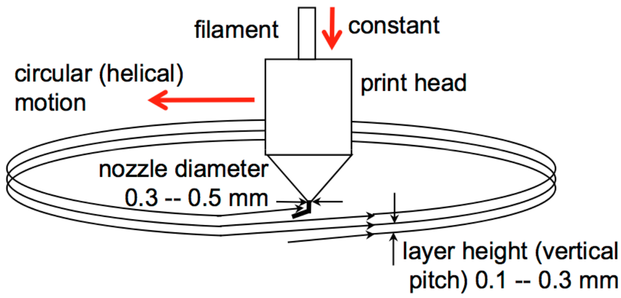
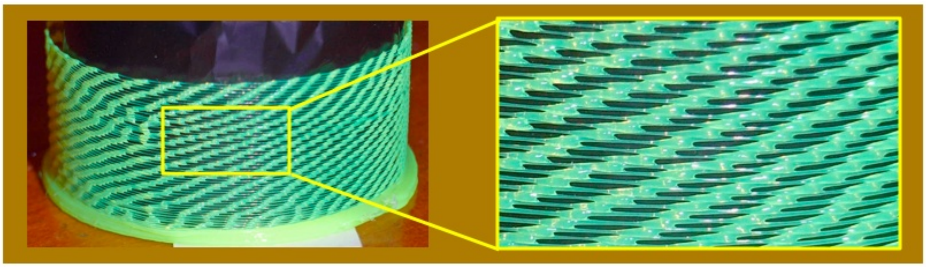


Fig. 2. 1D pattern generation method

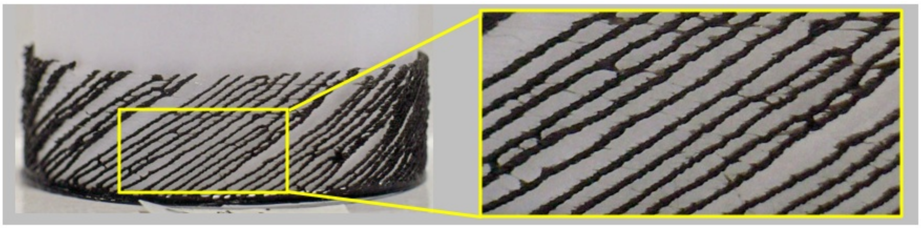
The important conditions and parameters for this method, which are constant, are as follows. The *nozzle diameter* is usually $0.5\,mm$ (0.5ϕ) or $0.3\,mm$, the average extruded *filament cross-section* (c) (which represents the velocity of extrusion) is much less than $0.2\,mm^2$ (0.5ϕ), and the *layer height* (h) is 0.1 to $0.3\,mm$ (in conventional 3D printing, the layer height is $0.4\,mm$ for a head with 0.5-mm nozzle). The number of line segments in a "circle" is 72. Other less sensitive parameters include the *filament material* (usually acrylonitrile butadiene styrene (ABS) or polylactic acid (PLA)), the *head temperature* (220 to 260°C for ABS and 180 to 220°C for PLA), and the head motion velocity (40 to $150\,mm/s$).

The pseudo layers are formed by using the following method. The usually-used initial state is all one (i.e., filled); that means, the first layer of the circles is fully (and slowly) filled with plastic. The second and above layers are printed using the above parameters. In the second layer, filament sticks to the first layer mostly periodically because the fluctuation is still small. However, upper layers may be less periodically because more fluctuations are caused. An example of the printing process can be seen in YouTube [Das 13].

Although several examples of printed results are shown in the previous study, typical patterns, which are shown in Fig. 3, are analyzed here at the first time. A Rostock MAX 3D-printer with a 0.5-mm nozzle and PLA filament was used for



(a) Stripes with thick ($h = 0.3\,mm$) layers ($c = 0.045$)



(b) Stripes with thin ($h = 0.1\,mm$) layers ($c = 0.02$)

Fig. 3. Typical printed patterns (by Rostock MAX)

printing them. Fig. 3(a) shows skewed stripes and strings generated by clockwise (right to left) head motion with 0.3-mm layer height. Counterclockwise head motion creates stripes skewed toward the opposite direction. This photo shows that stripes are generated by stacking chunks and the strings connect stripes. Fig. 3(b) also shows stripes and strings generated by clockwise head motion, but the layer height is $0.1\,mm$. The strings are very thin and mostly torn, so stripes are seldom connected by the strings.

3 Basic Simulation Method and Results

This section describes a computational model to simulate the self-organizing printing process, and shows relationships between printed patterns and simulation results.

3.1 CA-based simulation method

Figures 4 and 5 shows a computational model and the whole algorithm for simulating the printed patterns. Figure 4 shows the model, which is based on 1D asynchronous CA [Ing 84][Hof 87][Wik] (so the circle is quantized). This model simulates chunks but does not simulate strings. The values of cells, which grow upward, are calculated sequentially along the circle; that is, the value of one cell is decided in each step. Filament is constantly extruded in each step and it is accumulated until used for the cell (i.e., the cell value becomes 1). In the basic model, the accumulated filament is cleared when it is used; however, this rule may be varied. The value of the cell is probabilistically decided, and the filament is used only when it is sufficiently accumulated.

The value of a cell at layer l and location i (which is along the head motion) is defined by the following rule:

```
if cell[l-1][i] = 1 then cell[l][i] = 1 at probability p0
else if cell[l-1][i+1] = 1 then cell[l][i] = 1 at probability p1
else cell[l][i] = 0
```

Here, cell[l][i] means the value of cell at layer l and i-th location on the circle. The value of each cell is on (1) or off (0). Chunk-stacking probabilities $p0$ and $p1$ represent the explicitly-introduced randomness and decide the lifecycle and directions of stripes. The whole algorithm is described in Fig. 5. Instead of using a two dimensional array, which was used in the previous study [Kan 14], this algorithm uses a single-dimensional array for representing all the layers of cells because it is more convenient for simulating a helical motion.

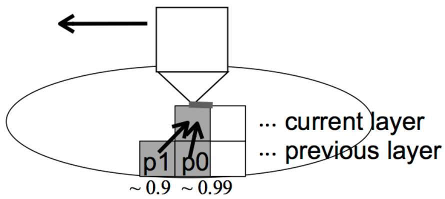


Fig. 4. Cellular automata for the simulation

3.2 Simulation of typical patterns

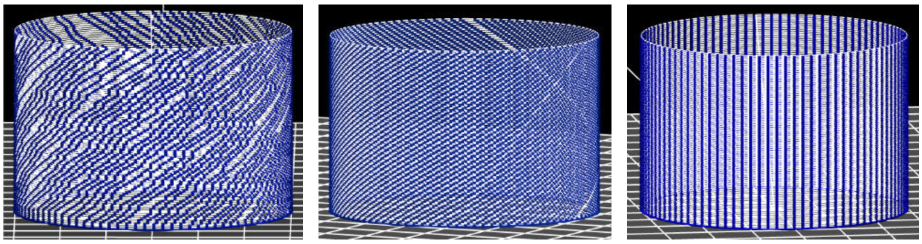
Several results of simulating typical patterns using a program based on the algorithm are shown in Fig. 6. The simulation program that generates G-code, which is a type of computer-aided manufacturing (CAM) programs, was written by Python. The resulting G-code programs were visualized by a CAM tool called Repetier-Host. Several simulation results of typical patterns with skewed stripes are shown in Fig. 6(a) and (b). The value of $p0$ must be less than 1 but it must be close to 1 to generate long-life patterns. If $p1$ is less than 1, a noisy pattern such as shown in Fig. 6(a) is generated. If $p1$ is 1, a crisp stripes as shown in Fig. 6(b) is generated. If $p0$ equals to 1, a vertical stripes as shown in Fig. 6(c) is generated. It is difficult to generate such patterns by printing, so they are not typical. However, sometimes they are generated under unknown conditions. An example of printed vertical patterns is shown in the appendix (Fig. 13).

```
N = 4 * 72; // number of arcs
Degree1 = 2 * pi / N; // degrees of an arc
extrudedFilament = 0;
for i in N, N + 1, ..., layers * N loop // repeat for all arcs
  cell[i] = 0;       // clear cell
  if extrudedFilament >= 1 then // Amount of extruded filament is sufficient.
    if cell[i-N] > 0 and random() <= p0 or      // check the cell below
      cell[i-N+1] > 0 and random() <= p1 then   // check the next cell
      cell[i] = 1;              // fill the cell
      extrudedFilament = 0.0;  // clear extruded filament
    end if
  end if

  angle = Degree1 * (i % N);    // "%" means modulo
  x = Radius * cos(angle);
  y = Radius * sin(angle);
  z = LayerHeight * i / N;
  drawNextArc(cell[i], x, y, z); // draw a cell (an arc)

  extrudedFilament = extrudedFilament + e1;  // extrude unit (0 < e1 < 1)
end loop
```

Fig. 5. Basic simulation algorithm



(a) Skewed stripes ($p0 = 0.99, p1 = 0.9, e1 = 0.4$)
(b) Skewed stripes ($p0 = 0.99, p1 = 1.0, e1 = 0.4$)
(c) Vertical stripes ($p0 = 1.0, p1 = 0$ to $1, e1 = 0.4$)

Fig. 6. Simulation of stripes

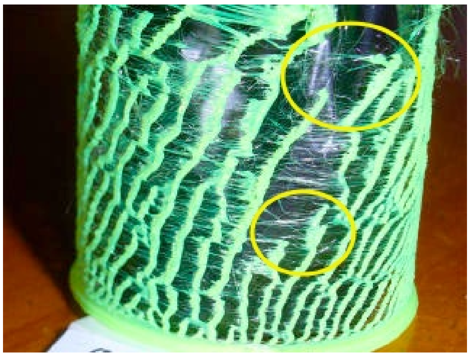
4 Various Types of Patterns in Printing and Simulation

Several types of printed patterns and simulation of the patterns are described in this section. The same printing parameter values as described in Section 2 were used unless otherwise stated.

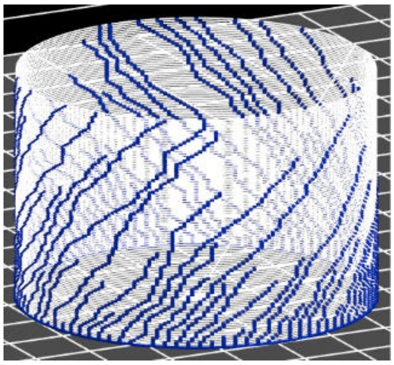
4.1 Extinction of stripes

Chunks sometimes failed to stick to chunks below, so the stripes may be extinguished. It is difficult to observe complete extinction in printed patterns because filament is constantly extruded. However, partial extinction is easily observed. An example is shown in Fig. 7(a). The circles show extinction of stripes. More extinction patterns are shown in the appendix (Fig. 14).

Extinction patterns can be simulated by making $p0$ smaller in the simulation program. Figure 7(b) shows an extinction pattern with $p0 = 0.97$.



(a) Print result (PLA, $h = 0.2$, $c = 0.02$, by Printrbot Plus)



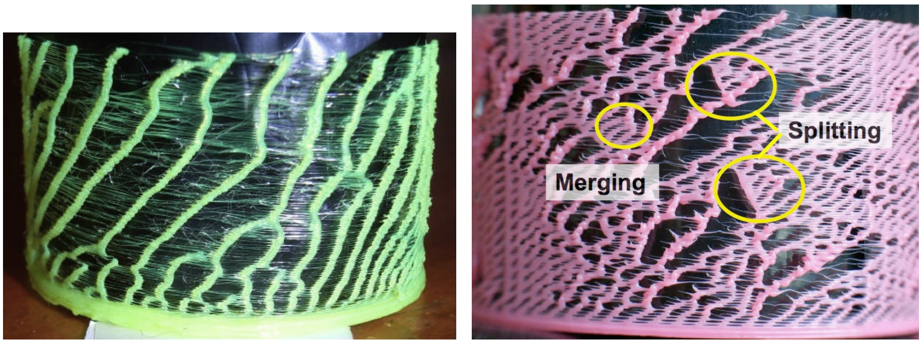
(b) Simulation result ($p0 = 0.97, p1 = 0.9, e1 = 0.6$)

Fig. 7. Extinction of stripes by printing and simulation

4.2 Splitting and merging stripes

Stripes are often split and merged. Fig. 8(a) shows a pure merging pattern and Fig. 8(b) contains both splitting and merging patterns. It is difficult to generate a pure splitting pattern. More splitting and merging patterns are shown in the appendix (Fig. 15). Note that vertical bars, which can be observed in Fig. 8(b), are caused by change of head velocity, which is caused by approximation of circles by linear lines. These vertical bars are artifacts that could not have been avoided yet.

The rule used for simulating splitting and merging patterns is described below and visualized in Fig. 9(a). The original computational rule is updated and two



(a) Pure merging pattern (PLA, $h = 0.2, c = 0.02$, by Printrbot Plus)

(b) Splitting and merging pattern (ABS, $h = 0.25, c = 0.045$, by Rostock MAX)

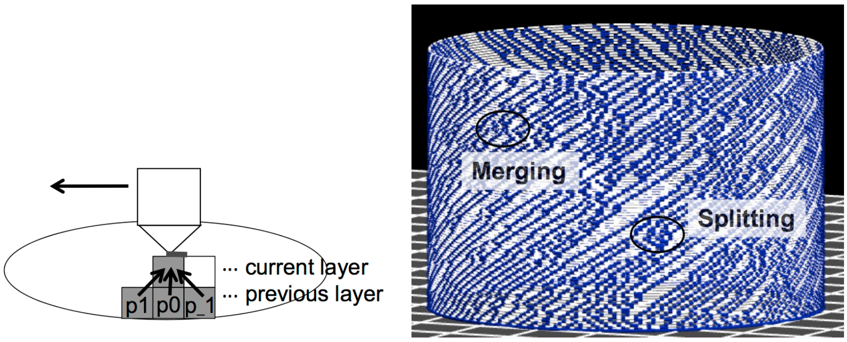
Fig. 8. Splitting and merging patterns by printing

more parameters, p_1 and C ($0 < C < 1$), are introduced because splitting and merging patterns cannot be simulated by the original algorithm, which never generates such patterns.

```
if extruded filament >= 1 then
  if cell[l-1][i-1] > 0 then
    { cell[l][i] = 1; clear filament } at probability p_1
  else if cell[l-1][i+1] > 0 then
    { cell[l][i] = 1; clear filament } at probability p1
  else if cell[l-1][i] = 1 then
    { cell[l][i] = 1; reduce filament by C } at probability p0
  else cell[l][i] = 0
else cell[l][i] = 0
```

The two new parameters are used in the following way. First, this rule introduces a dependence between cell$[l][i]$ and cell$[l-1][i-1]$, which enables splitting and which is controlled by a new chunk-stacking probability p_1. Second, the original rule always clears the extruded filament when it is used, but the new rule just subtract filament by C to preserve filament for splitting when the value of the cell below is 1.

Figure 9(b) shows a simulation result that contains both splitting and merging. The above rule can generate splitting-and-merging patterns; however, because the above rule modification is not the only way of introducing splitting and merging and the generated patterns look differently from printed patterns, a method for comparing the patterns and for evaluating the similarity should be developed, and the rule may have to be updated.



(a) Simulation method

(b) Simulation result ($p\_1 = 0.7, C = 0.8, p1 = 0.9, p0 = 0.995, e1 = 0.4$)

Fig. 9. Extended simulation method for splitting and merging and simulation result

4.3 Crossing waves

Patterns that look like waves often seem to cross stripes. Typical waves can be observed in Fig. 10. In this figure, waves can be observed by changes of stripe angles and by thick strings or absence of strings.

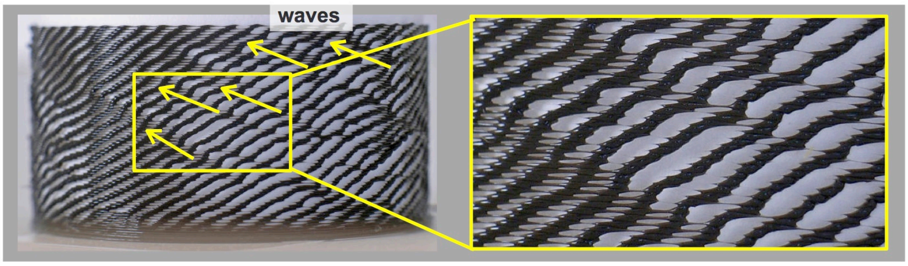
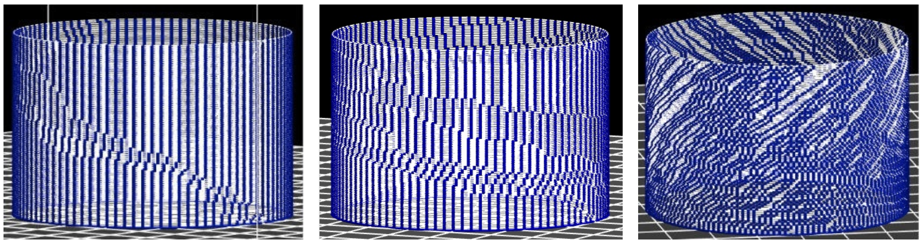


Fig. 10. Waves by printing ($h = 0.25$, $c = 0.045$, by Rostock MAX)

It is not possible to simulate the waves in Fig. 10 exactly by the proposed algorithm because the algorithm does not simulate strings; however, waves are considered to be propagation of some change or noise and can widely be observed in patterns generated by the algorithm or in CA in general. Such waves can be observed easier by slightly modifying a crisp result. For example, if there is a defect in a vertical stripes shown in Fig. 6(c), it is propagated such as the results shown in Fig. 11(a) or (b). Figure 11(a) shows another modified simulation result generated by the layered-circle-based (non-herical) algorithm in the previous study. In Fig. 11(b), zeros are randomly introduced to the initial layer (initial condition) as noises. Fig. 6(c) shows waves in a randomized simulation result. Vertical stripes are “propagated” nearly horizontally. Similar waves are also observed in Fig. 6(a).



(a) Propagation of defect in layered-circle model ($p0 = 1.0$, $p1 = 0$, $e1 = 0.4$)

(b) Propagation of explicitly introduced noises ($p0 = 1.0$, $p1 = 0$, $e1 = 0.4$)

(c) Waves in randomized model ($p0 = 0.99$, $p1 = 0.8$, $e1 = 0.5$)

Fig. 11. Waves by simulation using the original algorithm

4.4 Meshes

Stripes may sometimes be connected by layers of filaments, which may be called "meshes", such as shown in Fig. 12. Meshes may be caused by waves; however, the crossing lines of filaments seem to be different from thick strings in patterns with waves such as shown in Fig. 10. Thickness of crossing lines depends on the velocity of extrusion (i.e., the cross-section c). Meshes have not yet been successfully simulated by CA.

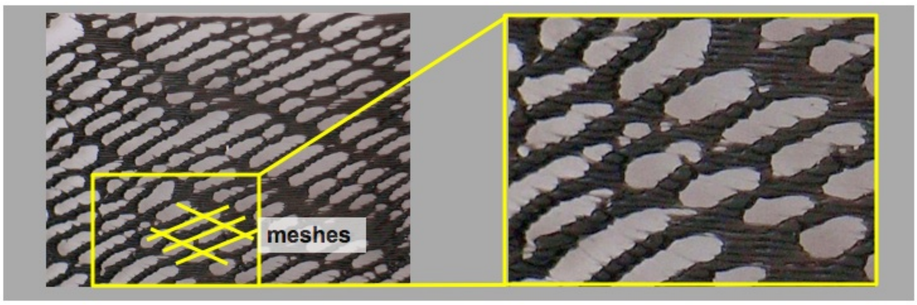


Fig. 12. Meshes by printing (PLA, $h = 0.15$, $c = 0.033$, by Rostock MAX)

5 Differences between Printing Process and Simplified Model

The printed and simulated patterns are different in the following three points. First, the computational model only simulates chunks and does not simulate strings. Especially, patterns with waves (and probably meshes) cannot exactly be simulated by this reason. Second, the width of printed patterns (of radius direction) varies, but it is not simulated. If the width becomes larger, the number of active (1) cells becomes smaller even if the amount of extruded filament does not change. (See Fig. 15(c).) Third, printed stripes may bend or oscillate when the print head comes but such motions are not simulated. There may be more differences.

6 Concluding Remarks

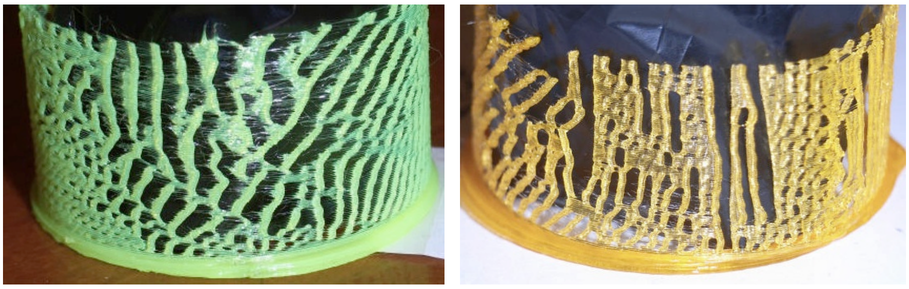
FDM 3D-printers can generate self-organized and "naturally-randomized" patterns, which consist of chunks and strings. Fully self-organized patterns can be generated by the proposed printing method and various types of patterns, i.e., parallel stripes, splitting and merging stripes, waves, and meshes, can be generated by using this method. These types of patterns can be partially simulated by proposed 1D-CA-based computational method. However, this method only simulates chunks but cannot simulate stripes, and there seem to be several differences between the printed and simulated patterns. These patterns should be compared by using a formal method and the CA-based model should be improved in future work.

References

[Das 13] Dasyn.com: Creating naturally-fluctuated patterns using a 3D printer. YouTube, http://youtu.be/IJ15ysJR5l8
[Gib 10] Gibson, I., Rosen, D. W., Stucker, B.: Additive Manufacturing Technologies. Springer (2010)
[Hof 87] Hofmann, M. I.: A Cellular Automaton Model Based on Cortical Physiology. Complex Systems 1, 187–202 (1987)
[Ing 84] Ingerson, T. E., and Buvel, R. L.: Structure in Asynchronous Cellular Automata. Physica D 10, 59–68 (1984)
[Kan 94] Kanada, Y.: The Effects of Randomness in Asynchronous 1D Cellular Automata. Artificial Life IV (1994) (Unpublished extended version, http://www.kanadas.com/CA/AsyncCA/AsyncCAext.pdf)
[Kan 14] Kanada, Y.: 3D Printing and Simulation of Naturally-randomized Cellular-automata. 19th International Symposium on Artificial Life and Robotics (AROB 2014) (2014)
[Rep] RepRap Wiki. http://reprap.org/
[Wik] Asynchronous cellular automaton. Wikipedia, http://en.wikipedia.org/wiki/Asynchronous_cellular_automaton
[Wol 84] Wolfram, S.: Universality and Complexity in Cellular Automata. Physica D 10, 1–35 (1984)

7 Appendix: More Printed Patterns

Various patterns has been generated by the printing method. However, limited number of printed patterns are shown above. Other patterns are shown in this appendix.



(a) PLA (by Printrbot Plus) (b) PLA (by Printrbot Plus)

Fig. 13. Vertical stripe patterns

7.1 Normal stripes

Vertical stripes easily occur in simulation as described in Section 3.2, but they are rare in print results. However, they occur in print results by a Printrbot-Plus 3D printer (Fig. 13). They have never be seen in print results by the Rostock MAX. They can be reproduced; however, no exact set of conditions that makes vertical stripes is known.

7.2 Extinction of stripes

Figure 14 shows an extinction pattern. (Figure 15(c) also contains extinction examples.) The ellipses show extinctions. Because extinctions generate surplus filament, thick chunks of filament are seen at the top.

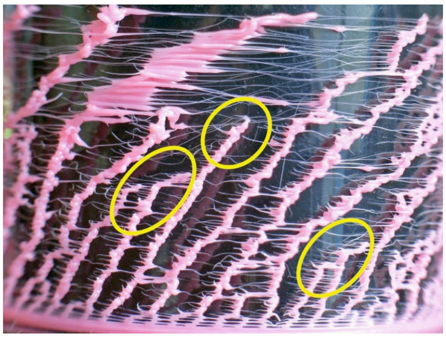
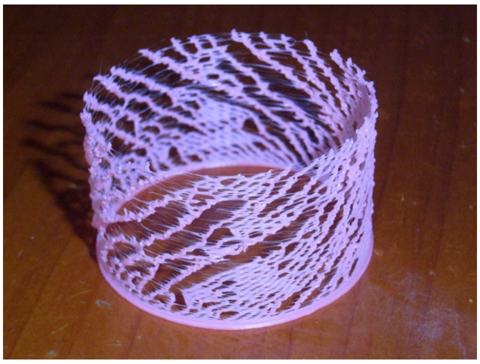


Fig. 14. More extinction patterns (ABS, by Rostock MAX)

7.3 Splitting and merging stripes

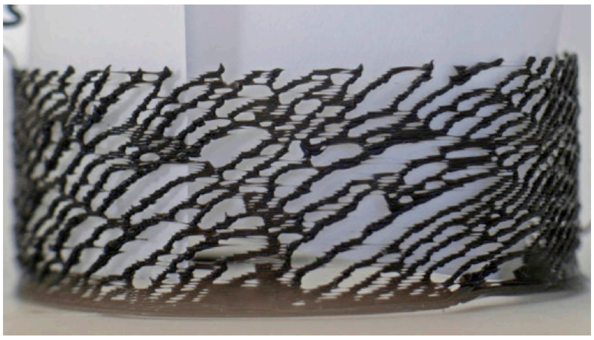
Complex patterns can be more easily generated by using ABS (Fig. 15(a) and (c)). Figure 15(c) shows a pattern that contains extinction, split, and merge.

The stripes at the top of this photo is very thick. Fig. 15(b) shows a pattern generated using PLA, which seem to contain splitting, merging, and waves.



(a) ABS (by Rostock MAX)

(c) ABS (by Rostock MAX)



(b) PLA (by Rostock MAX)

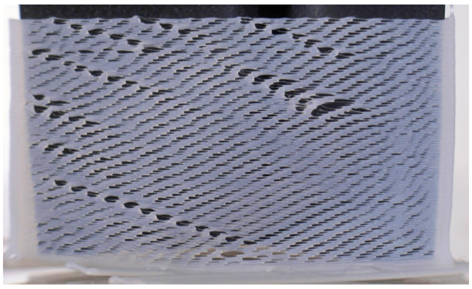
Fig. 15. More splitting and merging patterns

7.4 Crossing waves

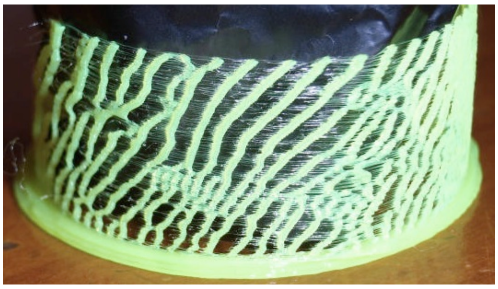
Wave-like patterns can be seen everywhere. Figure 16(a) is an example. However, a noticeable pattern shown in Fig. 16(b) is a combination of split, merge, and wave. At the center of this photo, stripes are split and merged. This pattern seems to propagate across the stripes.

7.5 Meshes

The mesh example shown in Section 4.4 contains thin meshes. More thick meshes can be observed in Fig. 17.

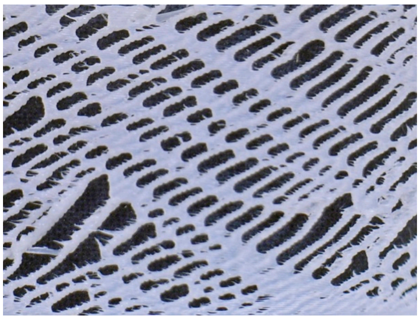


(a) ABS (by Rostock MAX)

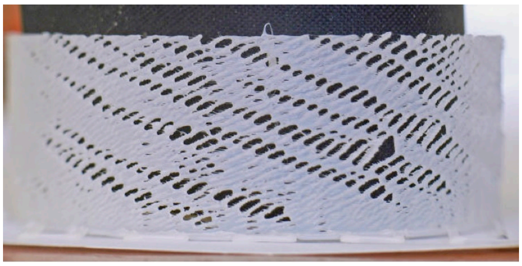


(b) PLA (by Printrbot Plus)

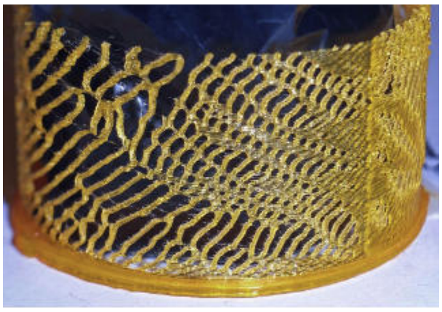
Fig. 16. More wave patterns



(a) ABS (by Rostock MAX)



(b) ABS (by Rostock MAX)



(c) PLA (by Printrbot Plus)

Fig. 17. More mesh patterns