Filling In: Livecoding musical, physical 3D printing tool paths using space filling curves

Evan Raskob
Goldsmiths, University of London
e.raskob@gold.ac.uk

Abstract

This paper explores how space-filling curves such as Hilbert Curves can be used for live, improvisational making in livecoding performances because of the predictable ways in which they fill up space without crossing over themselves. It explores current research into 2D space filling algorithms known as infills and how these relate to live, improvisational making. A livecoding system for 3D printers called LivePrinter is introduced and used to demonstrate different aspects of live performance with printers. These examples suggest some practical methods for working with Hilbert curves in a performance setting based, along with avenues for further research into livecoding performances using space-filling curves.

3D Printing as Performance

Livecoding can be a means for directing the physical movements of machines, such as robots and 3D printers. This change in media from sound and screen to machine presents some unique challenges. For 3D printing, one of the biggest challenges is how to improvise when making new forms out of extruded lines of plastic, without accidentally destroying those forms in the process. This means moving the printing tool head safely and without hitting previously constructed structures.

3D printing: a type of type of Computer Numerical Controlled (CNC) (i.e. digital) manufacturing technology called Additive Manufacturing (AM). More specifically, it is called Fused Deposition Modeling (FDM) or sometimes Fused Filament Deposition (FFD). FDM (as we will refer to it):

1. Is extrusion-based: physical objects built up by depositing layers of molten plastic material

2. Uses a motor to force a thin plastic filament through a hot extruder, melting it in the process (Turner 2014). Molten plastic sticks and bonds to previous layers, forming a solid

3. Is simple and safe enough process to be utilised and further adapted by maker spaces, universities and small businesses worldwide (Gao et. Al 2015)

4. Utilises a digitally-controlled print head or hot end encapsulating the extruder, moving in 3D space inside the printer cavity

5. Often requires 4 separate motors: Two motors position the print head relative to the print bed in x and y directions where x is side-to-side, y is front to back; one motor moves the print surface or print bed in the z direction, up and down. Another motor (e) feeds the plastic filament through the print head, also pulling it back at times to prevent unwanted material leakage in a process referred to as retraction.

As these motors spin, they vibrate and make sound. People have used 3D printers to make music, notably the Imperial March from Star Wars and, less notably, Nickelback (3D Print 2014). A library for the Python language is even available to convert MIDI note numbers to motor frequencies (Westcott, 2015). The sounds of the motors are relatively quiet but can be captured and then amplified using contact microphones and audio amplifiers. In a 3D printing livecoding performance, the performer choreographs the movements of the printer (speed, direction, duration) and the properties of the printer itself (temperature, fan speed, filament flow rate) by manipulating and writing code. Both the act of making, with its resulting physical forms, and the sounds of making are intrinsic to the performance. This leads to a dual mode of composition when making music for printers, where one can prioritise the aesthetics of the form by composing movements in millimetres or the aesthetics of the sound by composing in milliseconds of movement at specific speeds (e.g. musical notes).

X axis	Y axis	Z axis
47.069852	47.069852	160.0

Table 1.1: Typical speed scale for axis values for the motors used in the Ultimaker 2 printers, from Westcott's MIDI-TO-CNC library (Westcott 2015). Note that no values were given for the filament feeding (e-axis) motor.

The 4 digitally-controlled motors: they are the same model of motor and have identical properties. When the motors spin, they emit sound that can be mapped to notes in the equal temperament scale used by MIDI synthesizers using the following ES6-like pseudocode:

```
// calculate the frequency of the note from
// MIDI note number:

frequency = Math.pow(2.0, (note - 69) / 12.0)
    frequency = frequency * 440.0
// convert to motor speed in millimetres
// per second for GCode (see Table 1)

speed = frequency / speed_scale_for_axis;        (1)
```

Knowing the travel speed of each motor that will produces a desired musical note, along with the desired duration of that note, one can calculate the distance of travel across each axis by using a simple movement equation:

```
d = st              (2)
```

where d is the distance in mm to be calculated, s is a scalar representing the speed of the print head in mm/s in the current direction of travel, and t is the desired movement time in seconds.

In LivePrinter, this is simplified into two functions, one called m2s(NOTE) to translate a MIDI note NOTE to motor speed for an axis, the other called t2d(TIME) to use that speed and a desired duration of movement TIME to calculate distance. The following ES6 code uses these two functions to move the print head making a pitch of MIDI note C5 with a duration of 1 second (1000ms):

```
await lp.m2s(72).t2d(1000).go();               (3)
```

Normally, the speed at which the printer prints is determined by the desired quality of the print balanced by the total time of manufacture. Slower printing times generally lead to higher quality prints because the print head can follow a more precise path and the layers have time to cool properly before the next layer is applied on top (3D Matter 2015). When making music with printers, the relationship between form and making time is subverted. Performance time and musical quality can have as much weight as the quality of the finished object, whatever that final output might be in such a live setting. Slower speeds that might produce higher quality prints may be either inaudible or in the wrong key or frequency for the piece being performed.

This leads a maker/performer to reframe manufacturing as a mainly durational activity. Instead of describing objects in the usual way using technical drawings or digital models specifying physical dimensions in millimeters, one can consider objects using their durational dimensions and specify them in terms of the speed, angle and duration of movement used to manufacture them.

This tightly integrates the making of the object with the description of the object itself. It stands in opposition to the process planning approach that separates out a design concept from its fabrication processes.

Space Filling Curves Using Liveprinter

One solution to the problem of creating continuous, rhythmic movements in live 3D printing is a Hilbert curve:

- Originates with G. Peano (Peano 1890) &amp David Hilbert (Hilbert 1891; Sagan 1994).
- Curves map a one-dimensional space to a multi-dimensional space by passing through each point in that space once and only once.
- Generates tool paths which do not cross back over themselves
- Has a compelling visual aesthetic and mechanically rhythmic properties, as shown here from our experience performing with them
- Tool movements following a Hilbert curve that fill up spaces in predictable ways

Creating the curve

Curves can be represented as Lindenmeyer Systems (L-systems). There are ways to directly calculate a mapped point in n-dimensional space given an initial Hilbert index and a desired resolution.

These are less useful for describing tool paths because the information about how to move from point to point, i.e. the rotations and directions of movements encoded into an L-System string, are lost and must be re-calculated. The fact that an L-System encodes a literal set of movement instructions for a 3D printer, without need for matrix multiplications or any other form of interpolation, makes it attractive as a means for generating forms.

The LivePrinter example (Raskob 2019) uses a starting axiom of L and replacement rules of:

```
L: +RF-LFL-FR+          (4)
```

```
R: -LF+RFR+FL-          (5)
```

Finally, symbols are iterated in order and mapped to drawing functions written in ECMAScript 6 (ES6):

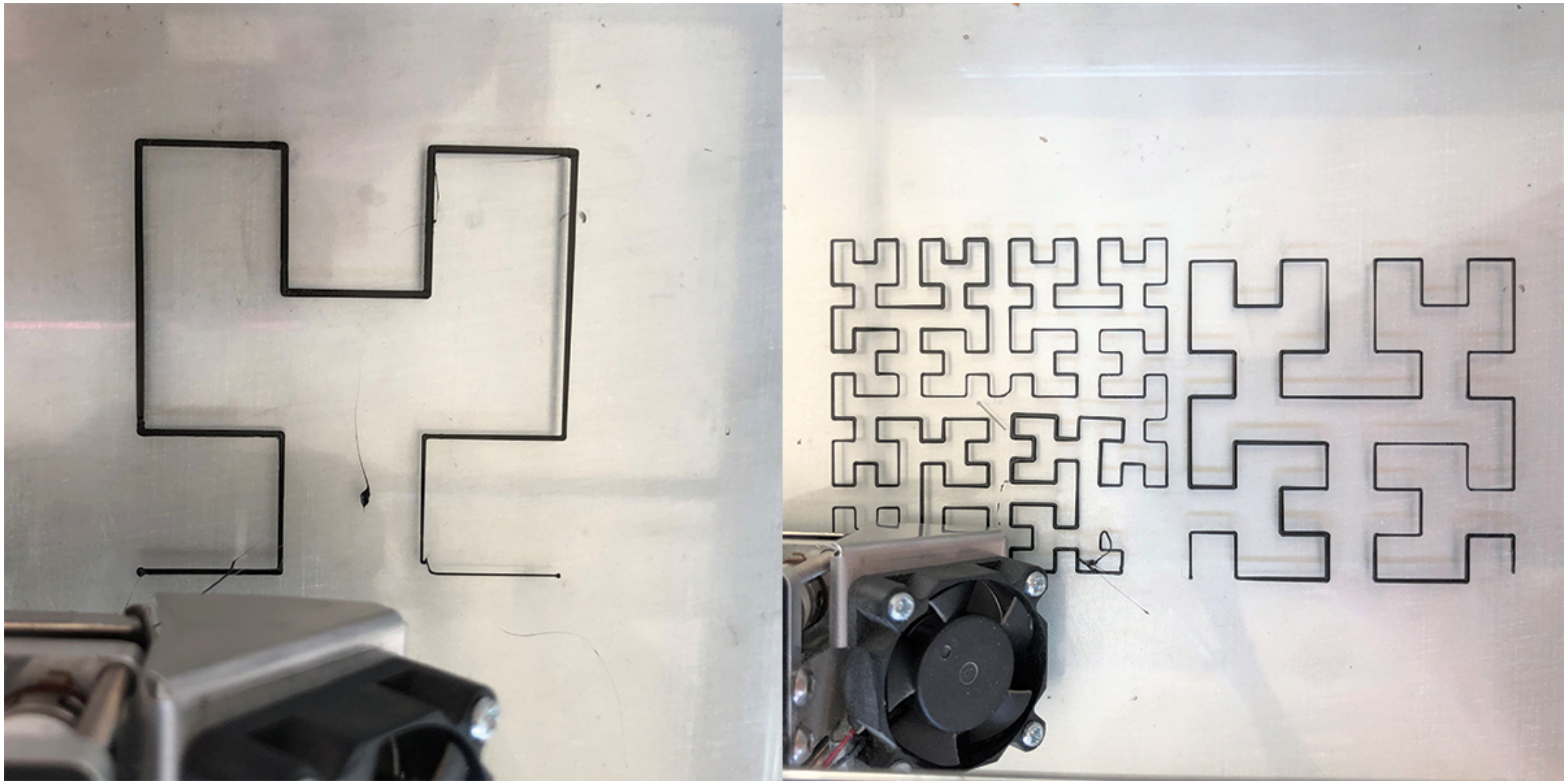

Figure 1: Three iterations of the Hilbert curve printed on a 3D printer. On the left is one iteration, on the far right is two iterations and on the middle right is 3 iterations. The code for generating these can be found in the LivePrinter repository `https://github.com/pixelpusher/liveprinter/blob/master/liveprinter/static/examples/hilbert.js`

```
'F': async () => { await lp.m2s(MIDI_NOTE).t2d(
    DURATION).go(1, false);
return 1; }
                                               (6)
'+': async () => { lp.turn(-90); return 0; }     (7)
'-': async () => { lp.turn(90); return 0; }      (8)
```

Note that:

- L and R symbols are ignored
- In (4,5,6), the lp object is an instance of the LivePrinter API object that converts drawing instructions to GCode
- In (6) m2s(MIDI_NOTE) is a function converting a MIDI note number to a motor speed as described above
- In (6) t2d(DURATION) is a function converting a movement duration in milliseconds to a movement distance in millimeters based on the motor speed as described above go(1, false) compiles the current drawing instructions to GCode and sends it to the physical printer, with the argument 1 meaning that the printer should be extruding during movements and the second argument false disabling retraction so the printer will not pause, retract, and then unretract after each drawing operation. lp.turn() is a virtual operation that rotates the current drawing direction by an amount in degrees, clockwise.

These drawing commands are similar to the transactional, embodied model of Logo's Turtle Graphics (MIT 2015). A drawing cursor (in this case, a 3D printer head) is directed to move and "draw" using short, intuitive statements that describe movement and drawing operations. A full API can be found in the documentation on the project website[1].

[1] https://github.com/pixelpusher/liveprinter

With LivePrinter, drawing a triangle with sides of 100mm could be achieved with the 3 lines of ES6 code:

```
await lp.turnto(0).dist(100).go(1);      (9)
await lp.turn(-120).dist(100).go(1);     (10)
await lp.turn(-120).dist(100).go(1);     (11)
```

This can be specified more concisely in the LivePrinter minilanguage as:

```
# turnto 0 | dist 100 | go 1             (12)
# turn -120 | dist 100 | go 1   (13)
# turn -120 | dist 100 | go 1   (14)
```

Of course, this assumes that the printer is prepared to draw with the print head having been heated up to the melting temperature of the plastic filament, the printing speed set, and the head positioned properly.

Filling Up Space With Plastic

Ordinarily the 3D printing process starts with a digital model created in a Computer Aided Design (CAD) program that defines the geometry of the object that is to be printed. This geometry mostly defines the outer and inner surfaces of the object, leaving the process of describing how the internals are to be manufactured to another piece of software called the "slicer".

The slicer determines the layer-by-layer construction of the object based on the properties of the model of printer to be used in that construction process. The process of starting with a model, optimizing the geometry, and preparing the machine instructions for manufacturing is often referred to as "Process Planning" (Thompson 2007; Livesu et. al. 2017).

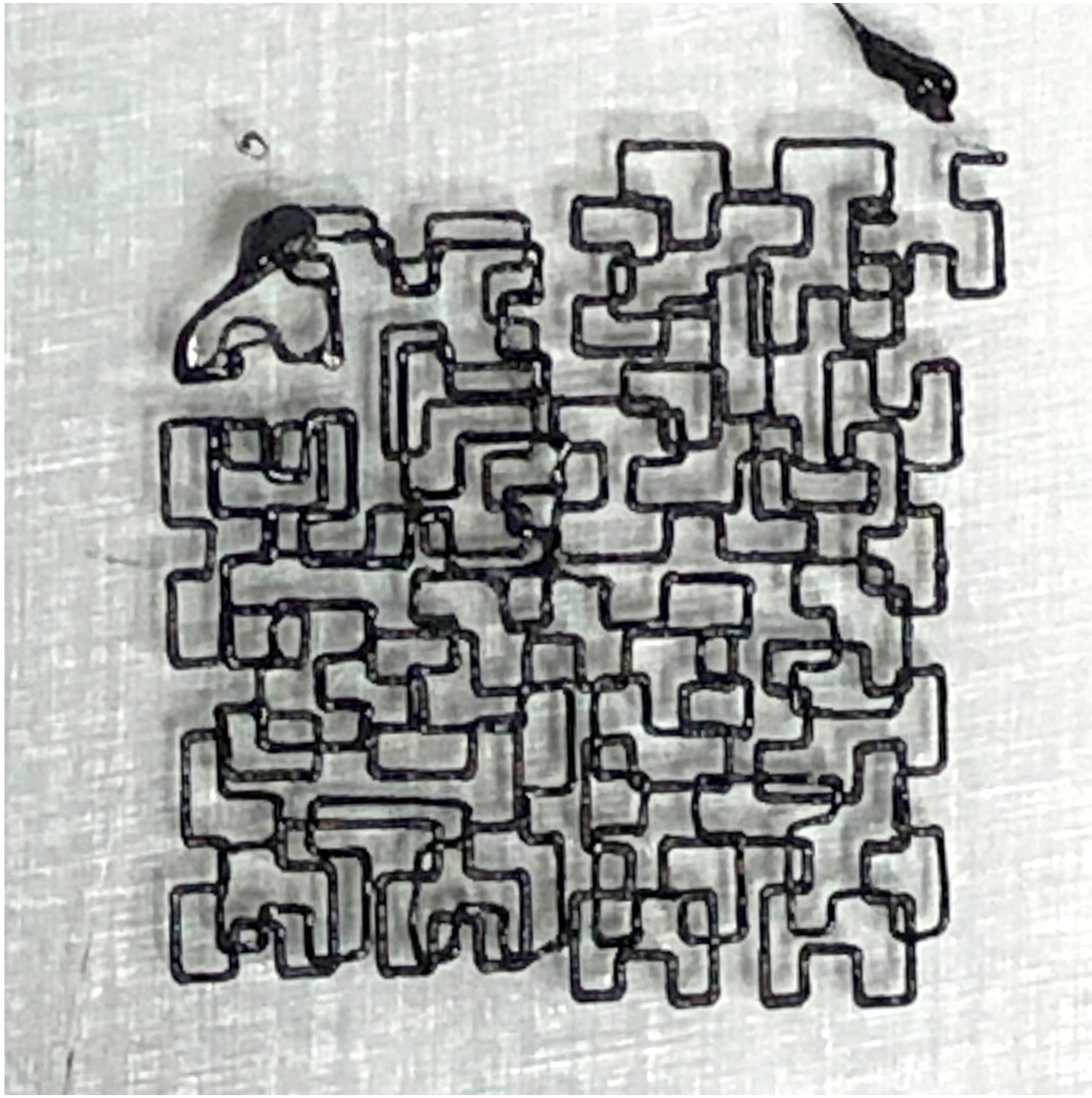

Figure 2: Two layers of Hilbert curve techno with random notes printed on a 3D printer. The code for generating these can be found in the LivePrinter repository `https://github.com/pixelpusher/liveprinter/blob/master/testing/hilbert_livecoding/hilbert-experiment-01.js`

The problem of how to efficiently generate 2D toolpaths that fill up 3D space in a structurally sound way whilst minimising time, material and movement remains an open area of research (Jin et. al. 2004; Jin et. al. 2017; Ding et. al. 2016). Current software packages offer a variety of filling patterns, from linear stripes to diamonds to vector patterns. Recent research has also looked at the properties of the Hilbert curve for filling spaces both rectangular and irregular (Ding et. al. 2016; Papacharalampopoulos et. al. 2018).

Motorised Hilbert Curves as Techno Perfromance

For 3D printing and music-making, the durations of movements and silences are of paramount importance to the musical aesthetics of the piece. As with any movement that creates music, a performer must control it precisely. Any gaps in movement or extraneous movements to position the printing head become part of the performance, for better or worse. A performer needs predictable tool paths at their disposal to improvise with, much as a jazz musician riffs on different musical scales and motifs. Continuous curves such as the Hilbert can be useful in that respect. As Papacharalampopoulos et. al. 2018 observed, Hilbert curves keep the print head moving throughout their length and thus minimize or altogether remove any extra waiting between operations and travel times needed to reposition the head after movements.

With dense Hilbert curves of higher orders, the number of movements can be quite large which is helpful to a livecoding performer trying to compose a dense melody live. They are especially useful when the printer movements are quick, giving them more time to think as they type new lines of code.

For example, a performer might wish to play a sequence of MIDI C5 notes (MIDI number 72) every beat, at a tempo of 120 beats-minute (or 0.5 seconds-per-beat) for a 12-beat segment. That means each beat the print head would be moving at a speed of 11.1165 millimetres-per-second for a distance of 5.5582 millimetres. A second-order Hilbert curve could completely contain that movement because the curve is made of 15 segments packed into a space that is 3 segments by 3 segments square, or 16.6747mm by 16.6747mm which will take a total of 6 seconds to play (12 notes at a duration of 0.5s per note). This whole curve fills up a little over 13% of the total printer bed space at that level, leaving room for about 8 repetitions of that motif before starting a new print layer.

Some calculations:

1. Printable area on the print bed is 200mm
2. 200mm/32 segments of 4th order Hilbert = 6.25mm per division
3. Gives us 1024 beats at 0.5 sec/beat –¿ 512 sec –¿ 8.533 minutes of music using the whole printer bed

Also, when the curve is oriented to the x/y axes of the print bed the tool head will move at right angles when drawing the curve. This uses only one motor at a time, playing distinct notes rather than chords. A performer could take advantage of this fact by alternating motor speeds with every segment of the curve, thus playing arpeggios. It is also possible to make more complex chords using rotated Hilbert curves. Rotating 45 degrees with respect to the x/y axes of the bed engages two motors simultaneously when moving, playing a two note chord consisting of the same two notes. A movement at any other angle produces a more complex tone that in practice can be difficult to control.

Future Challenges

In practice, it can be hard to manage Hilbert curves because the L-Systems representation of them grows exponentially with each iteration. This makes storing them as strings a non-trivial task, especially in a live, real-time performance setting.

Stepping through an entire curve in a performance is a dangerous task because it encompasses hundreds or even thousands of steps.

Some language-specific techniques are needed, such as using ES6 generators to write iterative Hilbert functions whose execution is not continuous (Mozilla 2019).

Additionally, more research is needed into the properties of other space-filling curves that have a number of segments that divide up evenly into common musical time signatures, like 4 and 8 beat segments. There are many other types of space filling curves that could have beneficial musical properties, including diagonal segments that represent chords.

Conclusions

Space filling curves can be a useful tool for live, improvisational physical making performances because of the ways in which they can completely fill up physical space with dense, aesthetically pleasing forms that do not cross over themselves during performance. Further study is needed to come up with methods for integrating different fill patterns over time, across unpredictable and improvised forms that arise during performance. This potentially has applications outside of sculptural performance, in the realm of industrial 3D printing, where 2D and 3D space filling patterns and techniques are current areas of research.

Acknowledgments

A special thank you to my PhD advisors, Prof. Mick Grierson and Dr. Rebecca Fiebrink, for all their support and feedback on this project.

References

Ding, Donghong, Zengxi Pan, Dominic Cuiuri, Huijun Li, and Stephen van Duin. 2016. 'Advanced Design for Additive Manufacturing: 3D Slicing and 2D Path Planning.' In New Trends in 3D Printing, edited by Igor V Shishkovsky. Rijeka: IntechOpen. https://doi.org/10.5772/63042

Gao, Wei, Yunbo Zhang, Devarajan Ramanujan, Karthik Ramani, Yong Chen, Christopher B. Williams, Charlie C. L. Wang, Yung C. Shin, Song Zhang, and Pablo D. Zavattieri. 2015. "The Status, Challenges, and Future of Additive Manufacturing in Engineering." Computer-Aided Design 69: 65–89. https://doi.org/https://doi.org/10.1016/j.cad.2015.04.001

Hilbert, D. 1891. Uber die stetige Abbidung einer linie auf ein Flaechenstueck. Math.Ann. 38, 459–460

Jin, Yuan, Yong He, Jian-Zhong Fu, Wen-Feng Gan, and Zhi-Wei Lin. 2014. 'Optimization of Tool-Path Generation for Material Extrusion-Based Additive Manufacturing Technology.' Additive Manufacturing 1-4: 32–47.

Jin, Yuan, Yong He, Guoqiang Fu, Aibing Zhang, and Jianke Du. 2017. 'A Non-Retraction Path Planning Approach for Extrusion-Based Additive Manufacturing.' Robotics and Computer-Integrated Manufacturing 48:132–44. https://doi.org/https:/doi.org/10.1016/j.rcim.2017.03.008

Livesu, Marco, Stefano Ellero, Jonàs Martínez, Sylvain Lefebvre, and Marco Attene. 2017. "From 3D Models to 3D Prints: An Overview of the Processing Pipeline." Computer Graphics Forum 36 (2): 537–64. https://doi.org/10.1111/cgf.13147

Matter, 3D. 2015. https://my3dmatter.com/what-is-the-influence-of-color-printing-speed-extrusion-temperature-and-ageing-on-my-3d-prints/

Matt Westcott. 2015. MIDI-TO-CNC. Retrieved Sept. 3, 2019 from https://github.com/gasman/MIDI-to-CNC/blob/master/mid2cnc.py

Milkert, Heidi. 2014. (Dec. 2014). 3D Printers Play Music from Mario Bros., Star Wars' Imperial March & More. Retrieved Sept. 3, 2019 from https://3dprint.com/29244/3d-printer-music-songs/

MIT. 2015. 'Logo History.' https://el.media.mit.edu/logo-foundation/what_is_logo/history.html%0A

Mozilla. 2019. 'Iterators and Generators.' https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Iterators_and_Generators

Papacharalampopoulos, Alexios, Harry Bikas, and Panagiotis Stavropoulos. 2018. "Path Planning for the Infill of 3D Printed Parts Utilizing Hilbert Curves." Procedia Manufacturing 21: 757–64.https://doi.org/https:/doi.org/10.1016/j.promfg.2018.02.181

Peano, G.: Sur une courbe qui remplit toute une aire plane. Math. Ann. 36, 157–160 (1890)

Raskob, Evan. 2019. LivePrinter; Livecoding for 3D printers. (Sept. 2019). Retrieved Sept. 3, 2019 from https://github.com/pixelpusher/liveprinter

Sagan H. 1994. Hilbert's Space-Filling Curve. In: Space-Filling Curves. Universitext. Springer, New York, NY

Sorensen, Andrew, Ben Swift, and Alistair Riddell. 2014. "The Many Meanings of Live Coding." Comput. Music J. 38 (1): 65–76. https://doi.org/10.1162/COMJ_a_00230

Thompson, Rob. 2007. Manufacturing Processes for Design Professionals. London: Thames & Hudson.

Turner, Brian N., Robert Strong, and Scott A. Gold. 2014. 'A Review of Melt Extrusion Additive Manufacturing Processes: I. Process Design and Modeling.' Rapid Prototyping Journal 20 (3): 192–204. https://doi.org/10.1108/RPJ-01-2013-0012

Zhao, Haisen, Fanglin Gu, Qi-Xing Huang, Jorge Garcia, Yong Chen, Changhe Tu, Bedrich Benes, Hao Zhang, Daniel Cohen-Or, and Baoquan Chen. 2016. 'Connected Fermat Spirals for Layered Fabrication.' ACM Trans. Graph. 35 (4): 100:1–100:10. https://doi.org/10.1145/2897824.2925958