

G-coordinator : G-code 生成の新たな手法とその可能性

G-coordinator: A new way to generate G-code

谷口 朝洋

Tomohiro TANIGUCHI

大阪府立大学

Osaka Prefecture University

【要約】

本研究は、3D プリンタを用いて造形を行う際に、機械に読み込ませるプログラムである G-code の新たな生成方法に関するものである。Python と数学的な関数を用いて G-code そのものを直接作成し、3 次元形状を構築することできる G-coordinator (<https://github.com/tomohiron907/G-coordinator>) というオープンソースソフトウェアを開発した。従来の「スライス」とは異なる手法によるパス生成により、より精密な印刷条件の調整が可能になるとともに、いままでは難しかった形状や、数理的な形状の容易な実現が期待できる。

キーワード: 3D プリンタ, G-code, ソフトウェア, モデリング

【Abstract】

This study focuses on the development of a novel method for generating G-code, the program read by a 3D printer. We have developed an open-source software called G-coordinator, which utilizes Python and mathematical functions to create G-code directly and construct three-dimensional shapes. This alternative path generation technique, which is distinct from traditional slicing methods, offers the potential to tackle complex shapes that were previously challenging to produce and enables the formation of more precise printing conditions and paths. It also facilitates the straightforward realization of mathematical shapes.

Keywords: 3D printer, G-code, software, modeling

1. 序論

一般的な G-code 生成は、CAD(Computer-Aided-Design)を用いて 3D データを作成し、それを stl データとして保存、その後、別のスライスソフトで stl データをスライスすることにより、G-code データを生成する。この場合、ノズルパス(経路)の生成はすべて機械任せとなり、パスの調整、編集は困難であった。

G-code はその大部分が、ノズルの移動していく 3 次元の座標列である。G-coordinator は、大量の座標列をユーザが認識、編集しやすいよう、その枠組みを提供する。それにより、複雑なパスのモデリングと、その精密な調整の両立を実現している。

CAD を介さずに G-code を生成するソフトウェアとして、Rhinoceros¹ のプラグイン Grasshopper² や、Excel 上で動作する FullControl GCode Designer³ などがある。これら

のソフトウェアに比べ、G-coordinator が持つ代表的な訴求点として、以下のものが挙げられる。

- オープンソース(MIT ライセンス)
- Python を使用して造形を行うため、様々なライブラリを活用でき、拡張性が高い。
- 多軸(5 軸, 6 軸)の 3D プリンタにも対応している。
- パスに対して、個別に印刷設定を調整可能
- インフィルや、stl のスライスなど、従来のスライスソフトの持つ機能も完備

2. G-coordinator の概要

G-coordinator は現在、Mac, Windows に対応している。以下の URL から、最新版実行ファイル(.app, .exe)の圧縮ファイル(.dmg, .zip)をダウンロードできる。(<https://github.com/tomohiron907/G-coordinator/releases>)

Python の実行環境があれば、src/main.py を実行して起動することができる。その際には、requirements.txt にまとめられたライブラリのインストールが必要である。

G-coordinator でのモデリングのスクリーンショットを図 1 に示す。G-coordinator の作業画面は大きく3つに分けられる。左から順に、エディタ、グラフィックスビュー、パラメータツリーの三つである。エディタで編集した python ファイルは、左上の実行ボタンを押すと、計算結果が 3 次元のパスとしてビューに描画される。形状を確認したのち、温度や速度などの印刷条件を調整し、右下のボタンを押すと、G-code が出力される。なお、ハードウェアのキネマティクスや、サイズ、スタート G-code などの設定に関しては、Machine settings から行う。より詳細な個別の印刷設定は、エディタから条件を指定すれば、グローバルな設定をオーバーライドできる。

3. G-coordinator におけるモデリング

G-coordinator におけるモデリングの方法を以下の円柱壁の例を用いて簡単に述べる。なお、円柱造形のサンプルコードを補足資料に掲載している。いかに複雑な形状であっても、基本的な造形のプロセスは同じである。3D プリンタのノズルから樹脂が吐出してできる一連の曲線(図 1 における白い円)を Path と呼ぶ。G-coordinator

では、このパスの x 座標列、 y 座標列、 z 座標列を計算して、Path オブジェクトを作成する。

```
for height in range(LAYER):
```

指定のレイヤー数だけループし、

```
arg = np.linspace(0, 2*np.pi, 100)
```

0 から 2π までの項数 100 の角度についての等差数列を作成し、

```
x = 10 * np.cos(arg)
```

```
y = 10 * np.sin(arg)
```

```
z = np.full_like(arg, (height+1) * 0.2)
```

その角度の \cos , \sin がそれぞれ x , y 座標列になる。 z 座標列に関しては、 \arg と同じ項数で、値がレイヤーごとに 0.2(レイヤー厚さ)大きくなる数列を用意すればいい。

```
wall = Path(x, y, z)
```

```
full_object.append(wall)
```

その後、 x , y , z 座標列を引数として、Path オブジェクトのインスタンスを生成する。生成したオブジェクトを full_object というリストに追加する。

object_modeling()という関数の返り値として、Path オブジェクトが要素であるリストが返されれば、G-coordinator は形状を認識し、計算可能である。

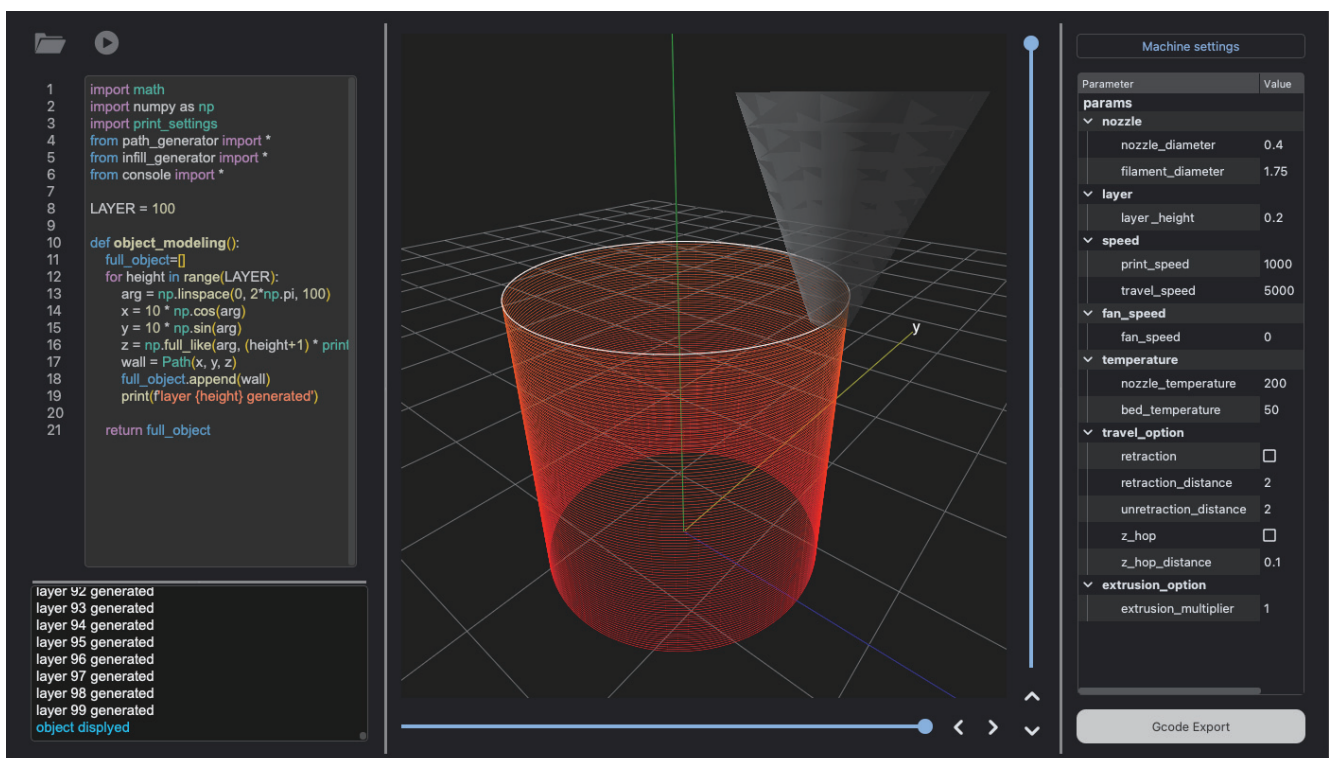


図 1 G-coordinator の操作画面

4. テストケース

以下に、G-coordinator を用いて、G-code を作成し、3D プリントしたテストケースを示す。いずれも、従来の CAD とスライス的手法では実現困難であった形状である。Github リポジトリの img フォルダにはその他の作例写真、example にはサンプルコードを 30 点近く公開している。<https://github.com/tomohiron907/G-coordinator/tree/main/example>)

- 網目テクスチャ

ノズルをジグザグに蛇行させ、一層ごとに位相を反転させることで、積層痕を意匠化させる技術。スライスソフトでは、直接ノズルパスを指定できないため、テクスチャを付与することが困難であった。

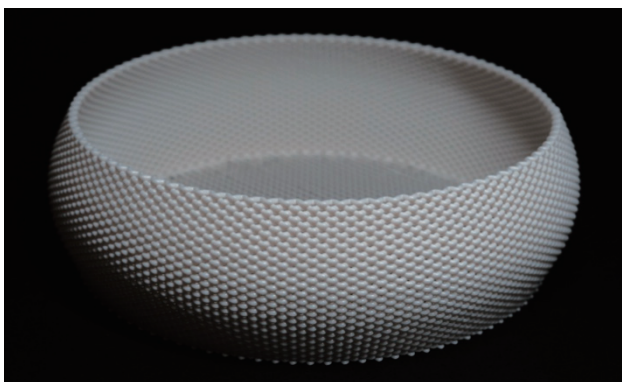


図 2 編み目テクスチャを持つトレー

- コッホ雪片

フラクタル形状を立体に押し出したもの。G-coordinator ではコードと関数を用いてモデリングを行うため、数理的な形状の実現が極めて容易にできる。他にも、一葉双曲面や、リサーチ曲線を造形するためのコードも公開している。



図 3 コッホ雪片の壺

- 振動による図柄表示

Python の持つ豊富なライブラリを用いれば、あらゆるものをモデリングのために活用できる。以下は、OpenCV⁴ のライブラリを用いて、モノクロ写真から、ある座標でのピクセルの濃さを抽出、その値を振動の振幅に重みとしてかけている。白のプレートを印刷後、黒の樹脂で振動パターンを印刷している。その結果、ミクロでは不規則な振動が、マクロでは写真に見える状況を実現している。



図 4 3D プリンタによる写真表現

- 多軸プリンタ

6 軸の 3D プリンタに対応した G-code も生成可能である。層を重ねるごとにノズルが傾いている。ハードウェアがまだ一般市場にないが、将来の 3D プリンタを見据えたソフトウェアとなっている。

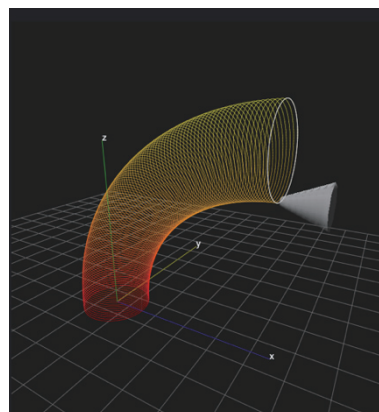


図 5 6 軸 3D プリンタの G-code 生成

5. 今後の展望

G-coordinator は、数学とコードにより、いままで出来なかった形状を実現する一方で、同時に、ユーザには多少の数学的知識と、プログラムに関するスキルを要求するものであり、参入のための障壁は比較的高いと言わざるをえない。そこで、現在想定している G-coordinator の将来像は大きく分けて二つある。

一つは、Processing⁵などを代表とするジェネラティブアートを実物化させるためのソフトウェアとしての活用である。これにより、アーティストやデザイナーは数学的なアルゴリズムを用いて、従来不可能だった美しい形状を3D プリンティングで具現化することが可能になる。

もう一つは、G-coordinator が数理的な形状に得意な性質を生かし、物理シミュレーションや計算力学を応用して形状を実物化させるソフトウェアとしての利用である。これにより、エンジニアや科学者は複雑な物理的な現象をシミュレートし、それを具現化するのに G-coordinator を活用できる。プログラムを使った3D形状の生成と制御が主眼となる点で、どちらの展望も共通している。

6. 結論

FFF 式 3D プリントユーザにとって、CAD などの造形手段と、印刷のための調整を行うスライサを同時に置き換える新たな手段としての G-coordinator のポテンシャルについて述べた。

スライスソフトに比べて、全部を機械任せにできない欠点はあるものの、裏を返せば、3D プリントの挙動の細部まで、マニュアル操作可能になると言える。

G-coordinator は、数学とプログラムの力を駆使して新しい形状を実現するための強力なツールとして、今後の3Dプリント分野でさらなる進化が期待される。

7. 参考文献

1. Robert McNeel & Associates, Rhinoceros, <https://www.rhino3d.com/en/>.

2. David Rutten at Robert McNeel & Associates. Grasshopper3D. <https://www.grasshopper3d.com>
3. Gleadall, A. (2021). FullControl GCode Designer: Open-source software for unconstrained design in additive manufacturing. Additive Manufacturing, Advance online publication. <https://doi.org/10.1016/j.addma.2021.102109>
4. OpenCV org. OpenCV Python. <https://github.com/opencv/opencv-python>.
5. Processing Foundation. Processing. <https://processing.org/>

8. 補足資料

以下に円柱を造形するためのサンプルコードを示す。

```
import numpy as np
import print_settings
from path_generator import *
from infill_generator import *
from console import *

LAYER = 100

def object_modeling():
    full_object=[]
    for height in range(LAYER):
        arg = np.linspace(0, 2*np.pi, 100)
        x = 10 * np.cos(arg)
        y = 10 * np.sin(arg)
        z = np.full_like(arg, (height+1)*0.2)
        wall = Path(x, y, z)
        full_object.append(wall)
        print(f'layer at {height=}')

    return full_object
```