Final Milestone for Toolpath Generation

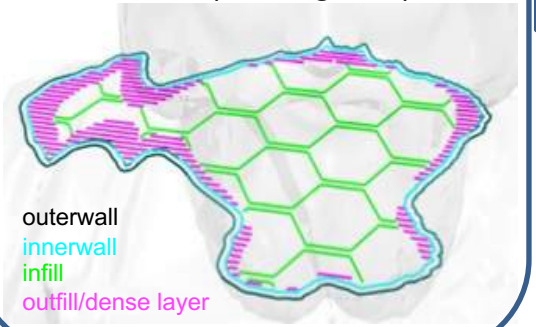
Nayoung Ahn and Dennis Diec

Task and aim

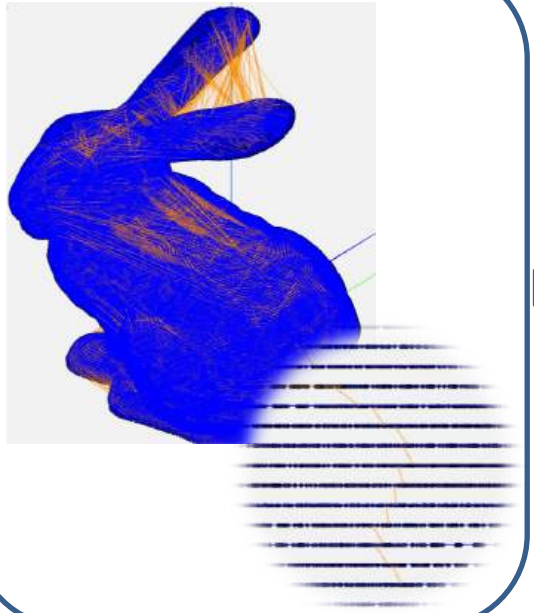
Read input

```
struct labelledLine{
        std::vector<PolyLine> pLine;
        LayerType pLineType;
        int layerIndex;
};
```

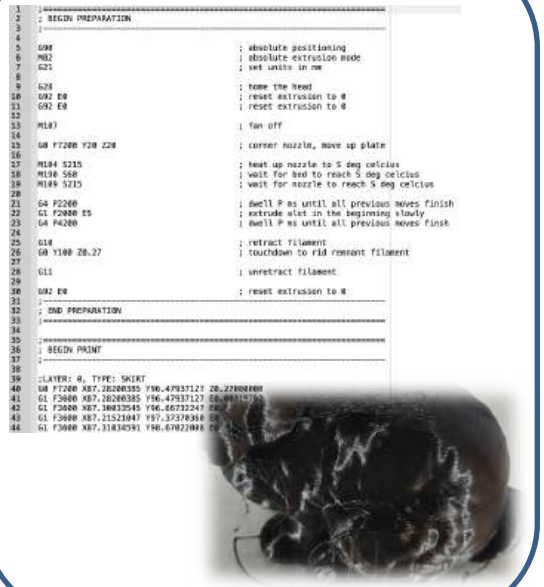
`LayerType` (i.e. regions)



Path planning



Print G-code



Timeline

Compile OpenFlipper with Plugin

- Understand:
  - G-code
  - Extrusion
  - OpenFlipper
- Generate simple snake print

First working version of Plugin

- Tune parameters in G-code
- Print based on perfect input
  - Single layer dense print
  - + Multilayer, multiplex
- Create diff. types of input

Final version of Plugin

- Path planning algorithm
  - Outer wall, Inner wall
  - Outfill / Dense layer
  - Infill
- Skirt generation
- Adaptive feedrate
- Improve UI and G-code print structure

G-code

G-code commands:

G## F## X## Y## Z## E##

M## S##

G-code Setting:

```
;=============================================================================
; BEGIN PREPARATION
;-----------------------------------------------------------------------------

G90                                     ; absolute positioning
M82                                     ; absolute extrusion mode
G21                                     ; set units in mm

G28                                     ; home the head
G92 E0                                  ; reset extrusion to 0
G92 E0                                  ; reset extrusion to 0

M107                                    ; fan off

G0 F7200 Y20 Z20                        ; corner nozzle, move up plate

M104 S215                               ; heat up nozzle to S deg celcius
M190 S60                                ; wait for bed to reach S deg celcius
M109 S215                               ; wait for nozzle to reach S deg celcius

G4 P2200                                ; dwell P ms until all previous moves finish
G1 F2000 E5                             ; extrude alot in the beginning slowly
G4 P4200                                ; dwell P ms until all previous moves finsh

G10                                     ; retract filament
G0 Y100 Z0.27                           ; touchdown to rid remnant filament

G11                                     ; unretract filament

G92 E0                                  ; reset extrusion to 0
;-----------------------------------------------------------------------------
; END PREPARATION
;=============================================================================

;=============================================================================
; BEGIN PRINT
;-----------------------------------------------------------------------------

;LAYER: 0, TYPE: SKIRT
G0 F7200 X87.28200385 Y96.47937127 Z0.27000000
G1 F3600 X87.28200385 Y96.47937127 E0.00319702
G1 F3600 X87.30033545 Y96.66732247 E0.01524222
```

First stage – simple geometries

Input: Created own polylines and with perfect ordering

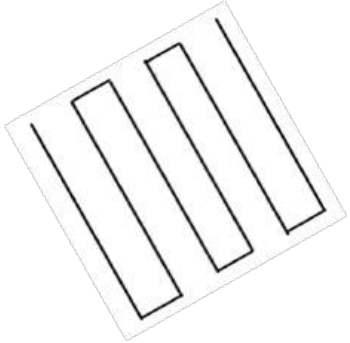
Method: `GenerateGCodeOwn`

- take any input and print from the first point to the next point for each layer

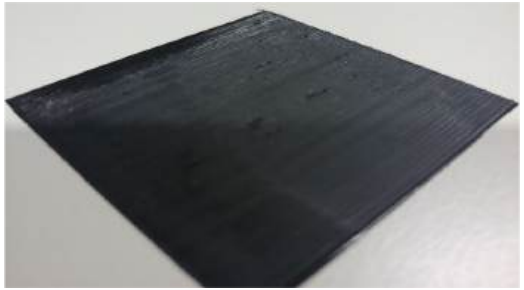
Observation:

- Good extrusion
- Basic understanding of G-code

Snake Line

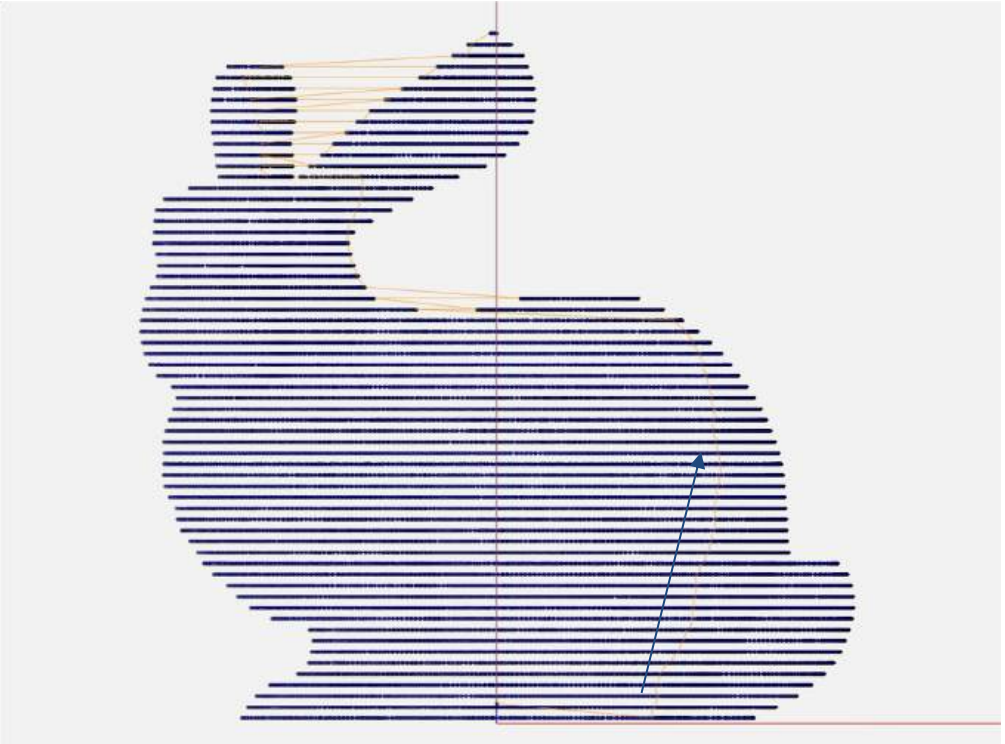


One dense layer



Five dense multiplex layers

Second stage – slices



(default size, setting)

Input: `std::vector<PolyLineCollection*> slicerings;`

Method: `GreedyAlgorithm`

- 1st outer wall, print sequentially
- Rest outer wall, print after reordering
  - the closest point may not be the first in the polyline

Observation:

- Good path planning
- Too high a layer height e.g. 0.4mm

Second stage – bad print

(size x: 30mm, default setting)

Input: `std::vector<PolyLineCollection*> slicerings;`

Method: `GreedyAlgorithm`

- 1st outer wall, print sequentially
- Rest outer wall, print after reordering
  - the closest point may not be the first in the polyline

Observation:

- Good path planning
- Too high a layer height e.g. 0.4mm

Third stage – outer, inner wall, and infill

Input: `std::vector<InfillGenerator::labelledLine> infillDataStruct;`

```
struct labelledLine{
       std::vector<PolyLine> pLine;
       LayerType pLineType;
       int layerIndex;
};
```

```
enum LayerType: int{
     OUTER_WALL = 0,
     INNER_WALL = 1,
     INFILL = 2,
     OUTFILL = 3,
     DENSE_LAYER = 4,
     SKIRT = 5
};
```

Method: `ToolpathAlgorithm`

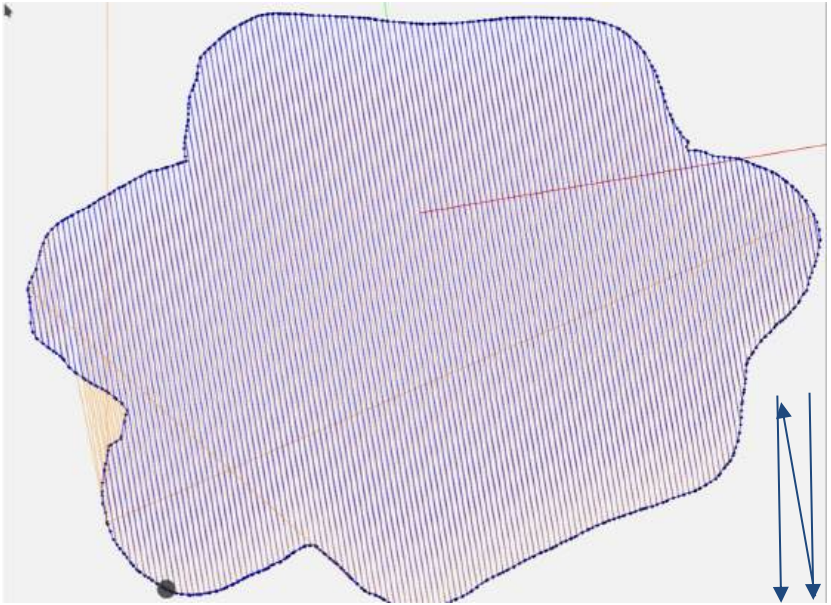
- Same as prior, but adapted to the input

Observation:

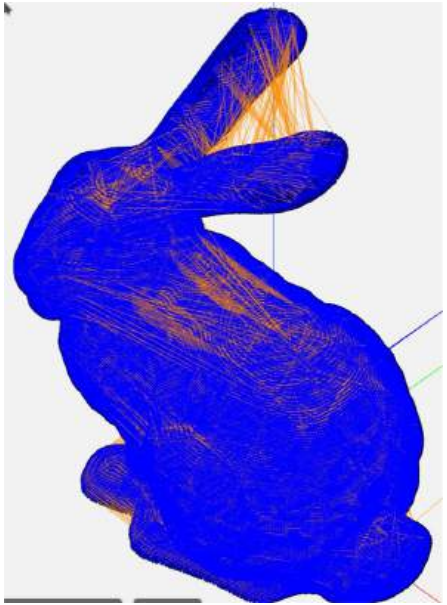
- Good extrusion
- Good layer height
- A lot of strings given crude path planning

(size x: 30mm, default setting)

Third stage – NC Viewer



First layer
`ToolpathAlgorithm` (size x: 60mm, default setting)



Entire bunny with infill
`ToolpathAlgorithm` (size x: 30mm, default setting)

Third stage – advanced algorithm

Input: same input

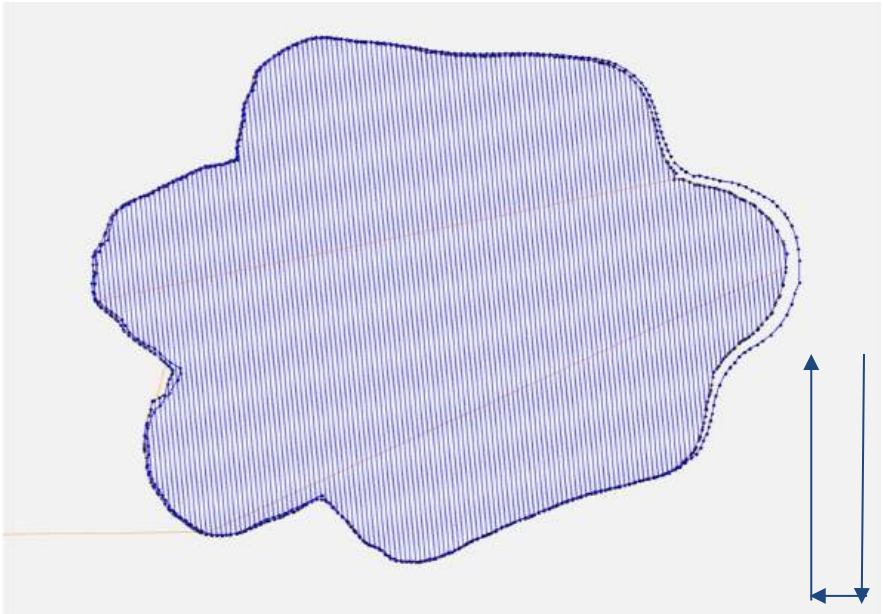
Method: `ToolpathAlgorithmAdvanced`

- Combination of 1st and 2nd Algorithm
- Also searches closest next polyline within layers

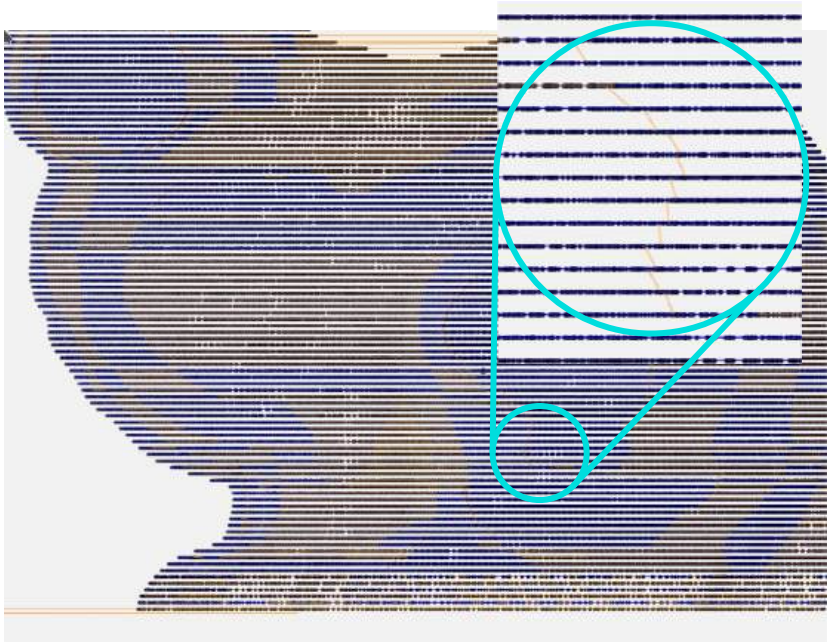
Observation:

- Still good layer to layer path planning
- Reduced strings due to improved path planning
- Bad intra layer path planning
- Broken function only works for certain values in setting (e.g.: size 60mm, layer thickness 0.4)

Third Stage – NC Viewer



First layer



Side view of entire bunny

`ToolpathAlgorithmAdvanced` (size 60mm, default setting)

Third Stage – bad print

Side view
`ToolpathAlgorithmAdvanced` (size x 60mm, default setting)

Top view of lower part of the bunny
`ToolpathAlgorithmAdvanced` (size x 60mm, default setting)

Third Stage – printing bunny with all components

Input: same input

- Empty polylines
- Multiple components (outerwall, innerwall, outerwall, innerwall, infillforall)

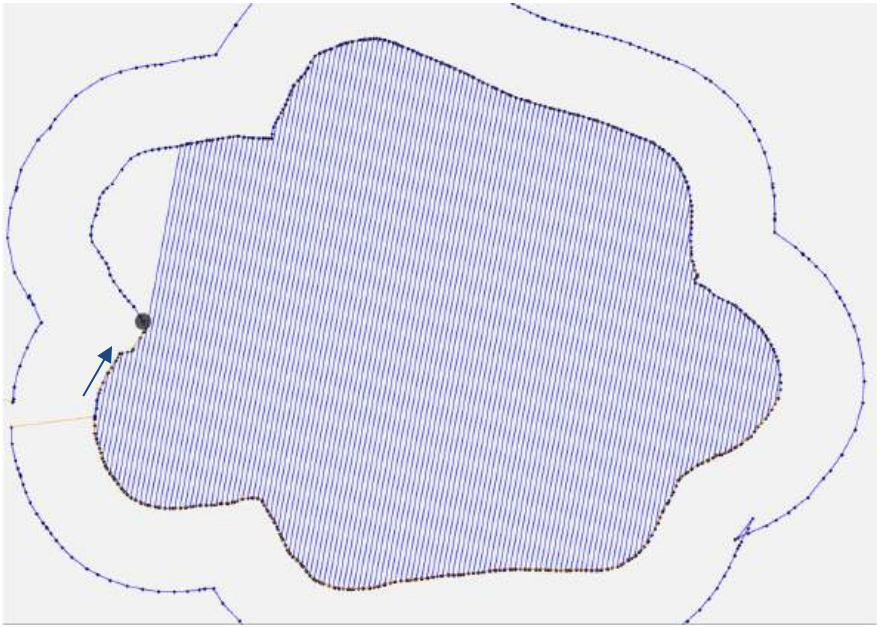
Method: `ToolpathAlgorithmExpert`

- Clear empty polylines
- Center the model
- Sort dense layer and infill
- Add skirt
- Apply adaptive feedrate
- Line segment check, boundary intersection check, finding corresponding indices
  - Intra layer path planning

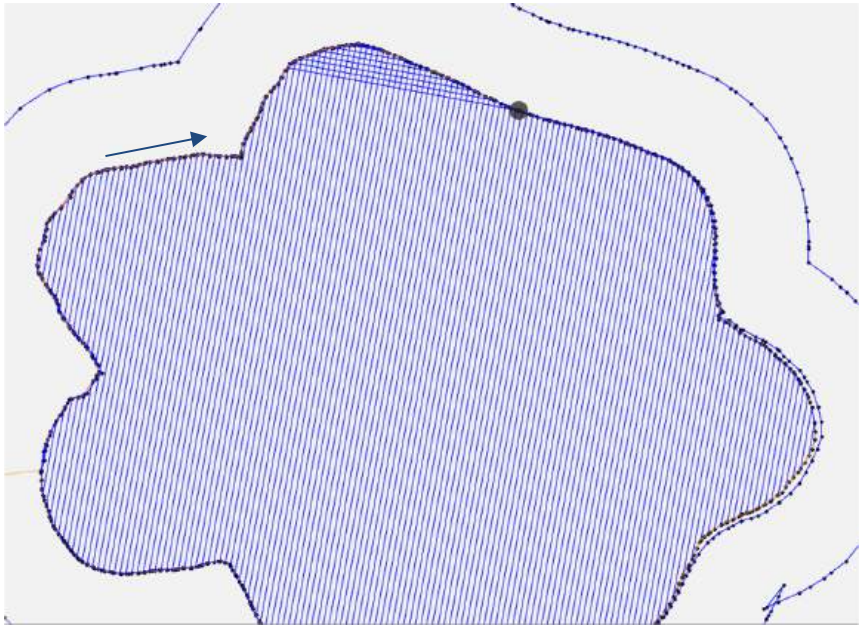
Observation:

- Expert path planning from layer to layer and within layer
- Missing dense layer at head and back
- Strings on multiple component

Third Stage – NC Viewer



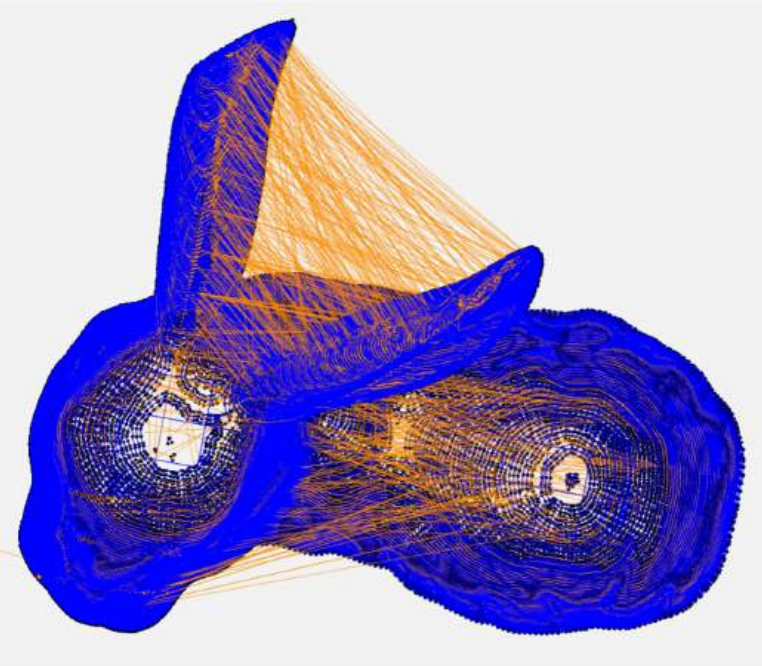
First layer



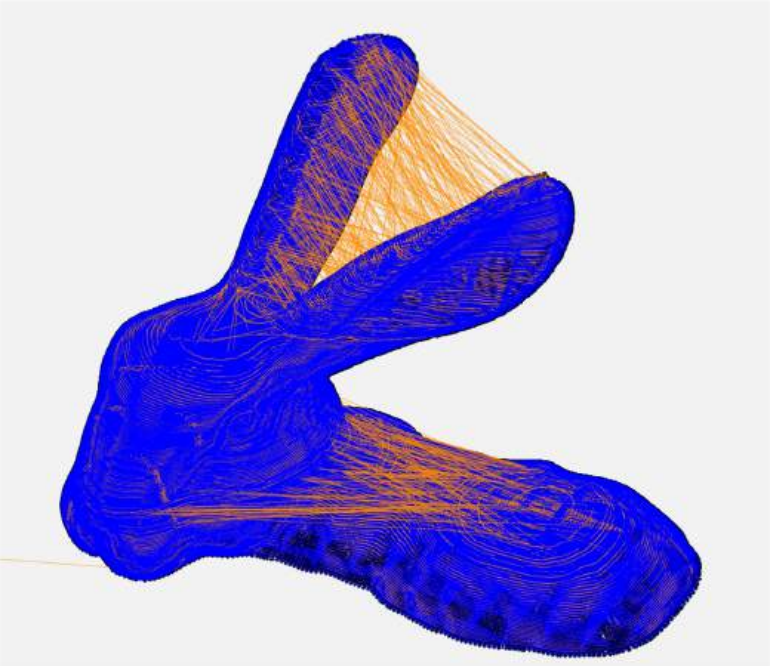
Second Layer

`ToolpathAlgorithmExpert` (size 60mm, default setting)

Third Stage – NC Viewer



Top view



Side view

`ToolpathAlgorithmExpert` (size 60mm, default setting)

Critical extrusion problem

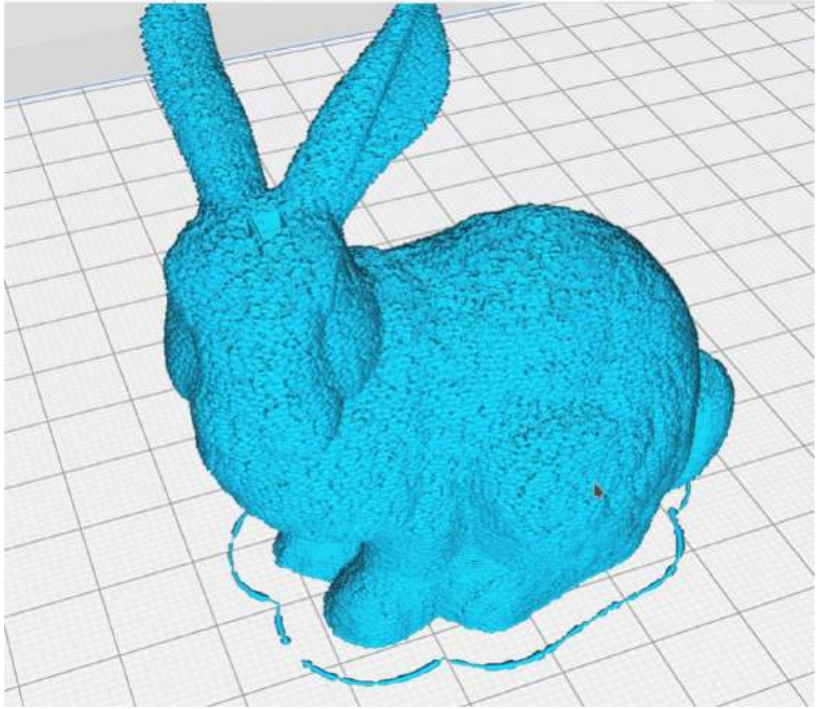
Feb 4, 2022 11:45pm GMT+0100

Commit abfa3c4f authored 5 days ago by

good night

```
QString("X%1 Y%2 E%3 \n").arg(curPoint[0], 8, 'f', 8, '0')
                         .arg(curPoint[1], 8, 'f', 8, '0')
                         .arg(extrusion_abs, 8, 'f', 8, '0');
```

E2951.69	→	E2951.69339377
E2951.70		E2951.70067477
E2951.70		E2951.70837599
E2951.71		E2951.71406599
E2951.71		E2951.71899687
E2951.72		E2951.72553430
E2951.72		E2951.72928101



Final product

Thank you for your attention