PL Techniques for 3D Printing

Chandakana Nandi
Anat Caspi
Dan Grossman
Zachary Tatlock

PAUL G. ALLEN SCHOOL OF COMPUTER SCIENCE & ENGINEERING
Compilers generate our environment.
Computerized Manufacturing

CNC
Computerized Manufacturing

CNC

Mold Making

Robotic Assembly
Computerized Manufacturing
Computerized Manufacturing
Computerized Manufacturing

CAD → IR → CAM → Machine Instructions
Computerized Manufacturing

- CAD
- CAM
- Machine Instructions
CAD/CAM: Idea → Part
CAD/CAM : Idea → Part

Where is the PL theory?
- semantics
- equivalence
- refinement
- approximation
CAD/CAM : Idea → Part

Where is the PL theory?
- semantics
- equivalence
- refinement
- approximation

Already worthy challenge, but recently…
Democratized Manufacturing
3DP: PL Opportunity

3D Printing Background

Challenge: CAD Synthesis

Challenge: Slicing Framework
Idea

3D Printing Workflow
1. Design

Idea

CAD

module snowman(scale, armAng) {
    rs = [scale, scale / 1.6, scale / 2.3]
    chopBase(0.65 * rs[0]) {
        sphere(r = rs[0]);
        translate([0, 0, 0.85 * rs[0]] +
            sphere(r = rs[1]));
        translate([0, 0, 0.85 * rs[1]] +
            sphere(r = rs[2]));
        translate([0, 0, 0.8 * rs]
            hat(scale);
        for() arm(scale, armAng);
    }
}

STL
1. Design

Idea → CAD

```
module snowman(scale, armAng) {
  rs = [scale, scale / 1.6, scale / 2.3]
  chopBase(0.65 * rs[0]) {
    sphere(r = rs[0]);
    translate([0, 0, 0.85 * (rs[0] +
      sphere(r = rs[1]));
    translate([0, 0, 0.85 * (rs[1] +
      sphere(r = rs[2]));
    translate([0, 0, 0.8 * rs
      hat(scale);
    for() arm(scale, armAng);
  }
}
```

STL

2. Slice

G-code

```
G1 X97.097 Y100.000 F6000
G1 E0.00000 F2400
G92 E0
G1 X97.239 Y99.103 E0.0136 F412
G1 X97.651 Y98.294 E0.0272
G1 X98.294 Y97.651 E0.0408
G1 X99.103 Y97.239 E0.0544
G1 X100.000 Y97.097 E0.0680
G1 X100.897 Y97.239 E0.0816
G1 X101.706 Y97.651 E0.0952
G1 X102.349 Y98.294 E0.1088
G1 X102.761 Y99.103 E0.1223
G1 X102.903 Y100.000 E0.1359
G1 X102.761 Y100.897 E0.1495
G1 X102.349 Y101.706 E0.1631
```
1. Design

Idea → CAD → STL

2. Slice

CAD → G-code

3. Print

Part → 3D Printer → G-code
1. Design

2. Slice

3. Print

4. OK?

3DP: PL Opportunity

3D Printing Background

Challenge: CAD Synthesis

Challenge: Slicing Framework
Challenge: CAD Synthesis

: CAD → STL

: STL → G-code

: G-code → Part
Challenge: CAD Synthesis

???

: Idea → CAD

: CAD → STL

: STL → G-code

: G-code → Part
Today: Crowdsourcing Designs

Internet → Idea → STL
Today: Crowdsourcing Designs

: Internet → Idea → STL

+ Easy
Today: Crowdsource Designs

: Internet → Idea → option STL

+ Easy

- Incomplete
Today: Crowdsource Designs

: Internet → Idea → option STL

+ Easy
- Incomplete
- Hard to modify
Today: Crowdsource Designs

Goal

For idea \( i \), even when:

\[
\text{Thingiverse} \ (\text{ }, i) = \text{None}
\]

there often exists similar \( i' \) such that:

\[
\text{Thingiverse} \ (\text{ }, i') = \text{Some } s
\]

So: adapt “almost” design \( s \) to a design for \( i \)!
Example “Almost” Designs

Rotated Hex Hole

Broken Chicken Legs
Inferring CAD to fix hex holder
Inferring CAD to fix hex holder

Bent wrench not seating parallel :(
Inferring CAD to fix hex holder

Bent wrench not seating parallel :(

Simple mesh editing broke model.
Inferring CAD to fix hex holder
Inferring CAD to fix hex holder

```diff
  difference() {
    cube([w, d, h]);
    for(i = [0 : len(holes) - 1])
      translate([offset(i), d/2, -1])
      hex_hole(holes[i]);
  }
```
Inferring CAD to fix hex holder

Infer (manual)

difference() {
    cube([w, d, h]);
    for(i = [0 : len(holes) - 1])
        translate([offset(i), 0, -1])
        hex_hole(holes[i]);
}

Tweak (small)

difference() {
    cube([w, d, h]);
    for(i = [0 : len(holes) - 1])
        translate([offset(i), 0, -1])
        hex_hole(holes[i]);

    if(i == 5)
        rotate([0, 0, 35])
        hex_hole(holes[i]);
}

hex_hole(holes[i]);
Inferring CAD to fix hex holder

Infer (manual)

Print (success)

Tweak (small)

difference() {
    cube([w, d, h]);
    for(i = [0 : len(holes) - 1])
        translate([offset(i), d/2, -1])
        hex_hole(holes[i]);

if(i == 5)
    rotate([0, 0, 35])
    hex_hole(holes[i]);

hex_hole(holes[i]);
Inferring CAD to fix chicken

Legs too thing => broke!

Expanding leg in STL tedious.
Inferring CAD to fix chicken

Infer (manual)

Tweak (small)

Print

(success)
Challenge: CAD Synthesis

decompiler

: Internet → Idea → CAD

: CAD → STL

: STL → G-code

: G-code → Part
3DP: PL Opportunity

3D Printing Background

Challenge: CAD Synthesis

Challenge: Slicing Framework
Exploring Slicing Strategies

Partitioning [Chopper 12]

Curved Layer FDM [CAD 08]

Parallelization

Multi-material [OpenFab 13]

Approx [Wireprint 14]
3DP Slicing Framework

Today: roll your own framework

Goal: LLVM / CIL for 3D Printing

Should enable exploring many new strategies:

- error compensation
- G-code synthesis with Z3
- parallelizing peepholes
- cross-part constraint
- …
3DP: PL Opportunity

Solid foundation:
- compiler theory
- fast solvers
- diverse synthesis
- num. methods

Goals:
- fab theory
- efficiency
- self-stabilize
- tools
Compilers generate our environment.
Compilers generate our environment.

PL folk can develop the tools to make it better.
Thank You!

Compilers generate our environment.

PL folk can develop the tools to make it better.