A Computer Engineering Approach To Design For 3D-Printing Manufacturability

Henry Dietz
Department of Electrical and Computer Engineering, University of Kentucky
`hankd@engr.uky.edu` `Aggregate.Org/hankd`

The popular press tends to describe 3D printers as crude prototypes of the replicators seen in *Star Trek*, constructing arbitrary things as complete devices including various active components and a wide range of materials. The truth is that these machines primarily make parts, and in most cases that is how it should be; various components, from microprocessors to simple screws, can be produced more effectively using other means. Beyond that, 3D-printed parts are subject to many machine-dependent manufacturability constraints; designing within such constraints is the primary impediment to many potential uses. Thus, we have been working toward automating design for 3D-printing manufacturability.

The main idea is that a design should not model an object, but be **a parameterized program describing the device in a structured, hierarchical, and composable way**. Such a program can be automatically transformed to better match the capabilities of the machines that will be used to produce parts – much as optimizing compilers rewrite a conventional computer program to better match the capabilities of the particular computer on which it will be executed.

Parametric Programmatic Tolerancing

As a simple example of programmatic design, consider specifying a statue with a removable rectangular base that is to firmly fit around the bottom allowing for printing tolerance. Using *OpenSCAD*, we could simply define an operator `module` to apply cylindrical tolerance offsets in `x`, `y`, and `z`:

```
module tol(xt=defxt, yt=defyt, zt=defzt) {
 for(c=[0:1:$children-1]) minkowski() {
  children(c); scale([xt, yt, zt]) cylinder();
} }
```

```
difference() {base(80); tol() statue(80);}
difference() {base(); tol() statue();}
difference() {base(); tol(yt=2) statue();}
```

Although *OpenSCAD*'s operator modules and inheritance rules are unusual, most modern CAD systems internally specify designs as programs that can be parameterized with comparable flexibility.

Manufacturability Issues

For any 3D printing technology, some structures simply cannot be reliably printed. For example, the Unified Thread Standard (UTS) thread profile specifies 30° angles that are too shallow to self-support on most extrusion-based consumer printers. Attempting to print the standard thread profile (shown blue external, red internal) results in drooping that makes the thread unusable. However, compatible 1mm pitch M42 lens adapters are printable on sub-$300 printers using layers as thick as 0.25mm simply by increasing the angle within the UTS part envelope:

A 45° angle is printable by nearly all printers, but the choice of angle is printer-dependent and even more precise threads can be obtained by accounting droop.

Similar design element/structure "substitutions" can enable making many complex objects. The tiny pliers (right) not only print assembled, but incorporate spanless hinge and spring in a single "metamaterial" part. Compiler technology can automatically recognize and transform programmatic designs in these ways.

Status And Future Work

Compiler transformation technology is easy for us. Creating the library of transformations is a slow and continuing process – collaborators welcome. See **`Aggregate.Org`** for details and other projects.