Quick Introduction to OpenSCAD

Joshua M. Pearce
Department of Materials Science & Engineering and
Department of Electrical & Computer Engineering,
Michigan Technological University, Houghton, MI, USA

OpenSCAD
The Programmers Solid 3D CAD Modeller

Michigan Tech
Michigan Technological University
Open Sustainability Technology Research Group
Make Everything Parametric

Allows later scaling, changing and newbie customization

All numbers should be made variables
Can use letters for simple designs // but comment
-advantages: simple equations
-disadvantage: big memory for large projects

Can use variable names describing it // box_length
-advantages: no comments, can read the code in English
-disadvantage: big messy equations
Design Using Primitive Shapes and Collecting Together

Simple → Complex
When Designing: Show Axes

Helps Orient Primitives
Know which way is up for printing!
a=5;
b=10;
c=20;
cube([a,b,c], center=true);
sphere(a, $fn=c);
//$fn is the resolution
cylinder(h = c, r1 = b, r2 = a, center = true);
Union Combining Primitives

“Try before you Buy”=%
union()
%cube([a,b,c], center=true);
sphere(a, $fn=c);
}
Difference - Subtraction

difference()
{
cube([a,b,c], center=true);
sphere(a, $fn=c);
}

Michigan Tech
Michigan Technological University
Open Sustainability Technology Research Group
Convex Hull of Child Nodes

```
hull(){
cube([a,b,c], center=true);
sphere(a, $fn=c);
}
```
Translate: Moving Stuff Around

union(){
cube([a,b,c], center=true);
translate([0,0,b])sphere(a, $fn=c);
}

Michigan Tech
Michigan Technological University
Open Sustainability Technology Research Group
Rounded Corners: Minkowski

$fn=50;$

minkowski() {
    cube([10,10,2]);
    // rounded corners
    cylinder(r=2,h=2);
}

Minkowski sums allow to add every element of A to every element of B.
Hand Crafting: Polyhydron

polyhedron ( points = [[0, -10, 60], [0, 10, 60], [0, 10, 0], [0, -10, 0], [60, -10, 60], [60, 10, 60]],
triangles = [[0,3,2], [0,2,1], [3,0,4], [1,2,5], [0,5,4], [0,1,5], [5,2,4], [4,2,3], ]);
Intersection : Keeps All Portions That Overlap

intersection() {
    cylinder (h = 4, r=1, center = true, $fn=100);
    rotate ([90,0,0]) cylinder (h = 4, r=0.9, center = true, $fn=100);
}

Michigan Tech
Michigan Technological University
Open Sustainability Technology Research Group
Make Each Completed Component a Module

Allows for more complex design

Clears the work space as modules are not shown unless called

Syntax:

```scad
module example(){ put your module scad here }
```

Call it by:

```scad
example();
```
module example()
union()
    cube([a,b,c], center=true);
    translate([0,0,b])sphere(a, $fn=c);
}

example();
Manipulate Your Module

rotate([45,0,0])example();

hull() {
example();
}

Add, subtract modules etc.
For Repetitive Tasks Use Loops

for (i = [1:12])
{
    assign (angle = i*30)
    {
        rotate(angle, [1,0,0])
        example();
    }
}
Putting it All Together to Make Something Useful

Shelling corn is a chore done by hand in much of the rural developing world. Yet there are handy corn shellers that can save people hours of labor. **DIY shellers are a big chore to make...so you can print one.**

The finished, cement-filled corn sheller is on the right. A commercial aluminum corn sheller is on the left. The bottom sheller that was cut from a PVC pipe cap. It did not perform as well as the can.
Step 1: Break Complex Object Into Simple Parts

- Bucket: 2 tapered cylinders
- Fingers: 2 hulled cylinders
  - Fingers tapered in

Consider improvements:
- Grips on outside – use fingers
Parametric – Design ALL of the Products at Once

Step 2: Lay out variables with comments to input to Customizer

```
//Open-source parametric hand corn sheller

// height of corn sheller
h=55;

// radius of top of corn sheller
rt=35; // [50:130]

rb=0.85*rt; // radius of bottom of corn sheller

// number of digits
d=6;

// digit radius
r=1.5;

// extra length of digit
l=1;

// thickness of sheller
t=3;
```
Using Modules

module sheller()
union()
for (z = [0:d]) // d iterations, z = 0 to d
{
    rotate([0,0, z*360/d])translate([rb,0,h*.1])finger();
}
difference()
{
cylinder(h = h, r1 =rt, r2 =rb, center = true, $fn=100);
    translate([0,0,0])cylinder(h = h+1, r1 =rt-t, r2 =rb-t, center = true, $fn=100);
}
}
module finger()
{
    rotate([0,(rb/rt)*-10,0])
hull()
{
cylinder(h = h*.9, r1 =2*r, r2 =2*r, center = true, $fn=100);
    translate([l-(rt-rb),0,0])cylinder(h = h*.9, r1 =r, r2 =r, center = true, $fn=100);
}
}

Best practices:
Indent to see,
comment everything
$fn=100; one time

Many Fingers
Bucket Wall
Finger
Parametric Corn Sheller
Thingiverse Customizer

Customizable corn sheller

Parameters

H height of corn sheller
6.5

Rt radius of top of corn sheller 50

D number of digits
4

R digit radius
2.5

L extra length of digit
-4

T thickness of sheller
4

http://www.thingiverse.com/app

Anyone can make a corn sheller perfect for them with no coding.
Use Past Work

Libraries:

use <MCAD/involute_gears.scad>
include <escapementLibrary.scad>

You are using collections of Modules written before...
### Cheat Sheet

#### Syntax

```plaintext
var = value;
module name(...) { ... } name();
function name(...) = ...
include <...scad>
use <...scad>
```

#### 2D

- `circle(radius)`
- `square(size,center)`
- `square([width,height],center)`
- `polygon([points])`
- `polygon([points],[paths])`

#### 3D

- `sphere(radius)`
- `cube(size)`
- `cube([width,height,depth])`
- `cylinder(h,r,center)`
- `cylinder(h,r1,r2,center)`
- `polyhedron(points, triangles, convexity)`

#### Transformations

- `translate([x,y,z])`
- `rotate([x,y,z])`
- `scale([x,y,z])`
- `mirror([x,y,z])`
- `multmatrix(n)`
- `color("colorname")`
- `color([r, g, b, a])`
- `hull()`
- `minkowski()`

#### Boolean operations

- `union()`
- `difference()`
- `intersection()`

#### Modifier Characters

- `*` - disable
- `!` - show only
- `#` - highlight
- `%` - transparent

#### Mathematical

- `abs`
- `sign`
- `acos`
- `asin`
- `atan`
- `atan2`
- `floor`
- `round`
- `ceil`
- `ln`
- `len`
- `log`
- `lookup`
- `min`
- `max`
- `pow`
- `sqrt`
- `exp`
- `rands`

#### Other

- `echo(...)`
- `str(...)`
- `for (i = [start:end]) {...}`
- `if (...) {...}`
- `assign (...) {...}`
- `search(...)`
- `import("...stl")`
- `linear_extrude(height,center,convexity, twist, slices)`
- `rotate_extrude(convexity)`
- `surface(file = "...dat", center, convexity)`
- `projection(cut)`
- `render(convexity)`

#### Special variables

- `$fa` - minimum angle
- `$fs` - minimum size
- `$fn` - number of fragments
- `$t` - animation step

---

http://www.openscad.org/documentation.html

---

**Michigan Tech**

Michigan Technological University
Open Sustainability Technology Research Group
Thank you!

More information

- http://www.openscad.org/
- http://www.appropedia.org/MOST
- http://reprap.org/
- pearce@mtu.edu