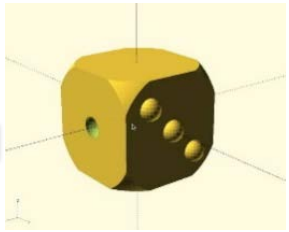
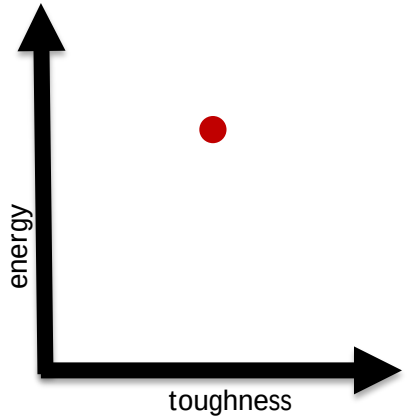
Parametric Modeling Basics

Wojciech Matusik

Computational Design Stack

```
26 ;Layer count: 336
27 ;LAYER:0
28 M107
29 G0 F9000 X91.800 Y93.520 Z0.300
30 ;TYPE:SKIRT
31 G1 F1200 X92.617 Y92.870 E0.01964
32 G1 X93.518 Y92.412 E0.03865
33 G1 X94.458 Y92.141 E0.05705
34 G1 X95.218 Y92.072 E0.07141
35 G1 X95.998 Y92.064 E0.08608
36 G1 X96.894 Y92.071 E0.10294
```





Hardware

Machine Code

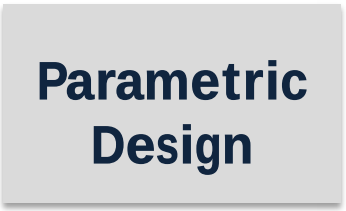
Direct Design

Functional Design

Parametric Design

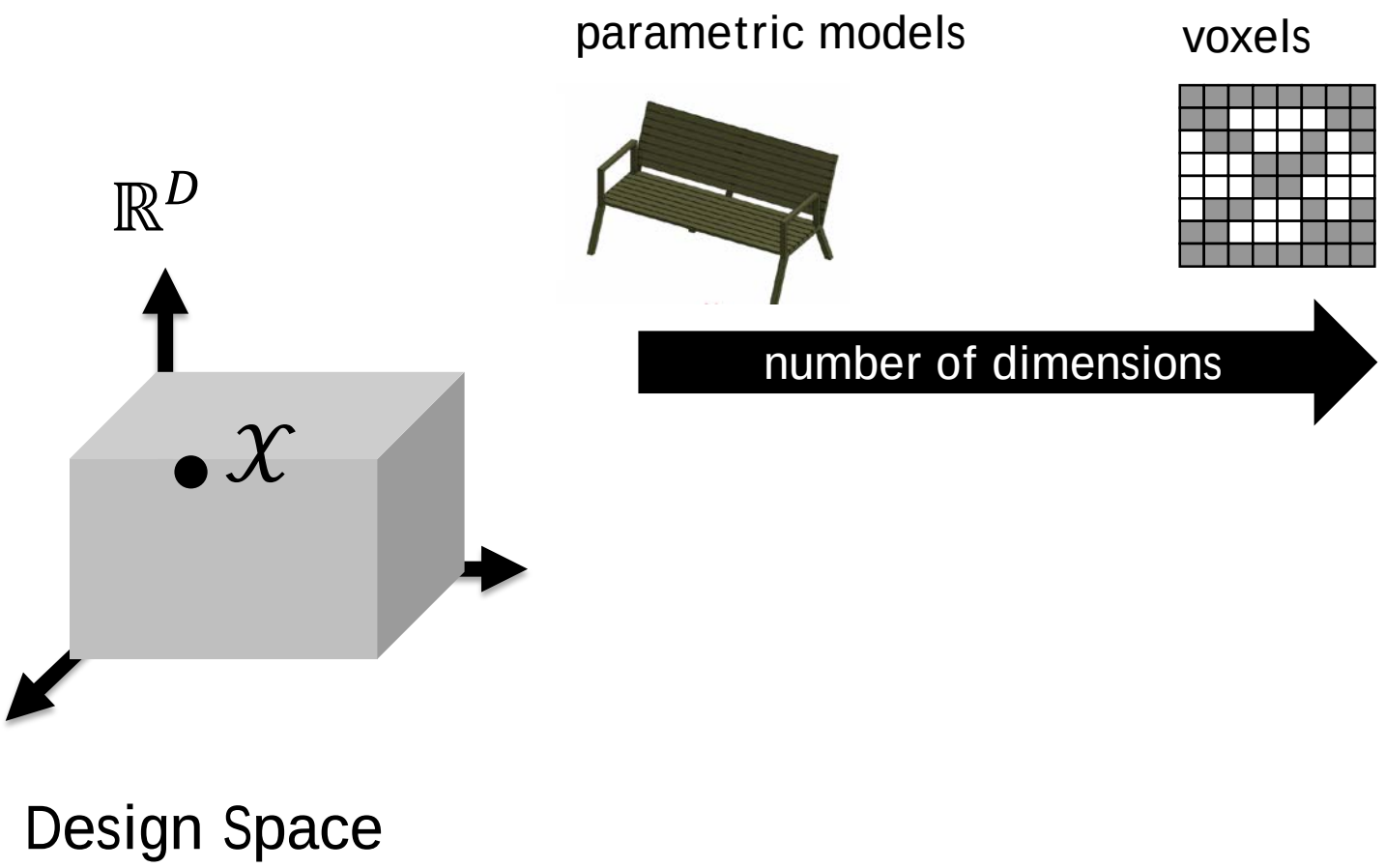
$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}$$

parameters



Design Space

- Each design can be mathematically represented as a point in $\mathbb{R}^D$



Why Parametric Design?

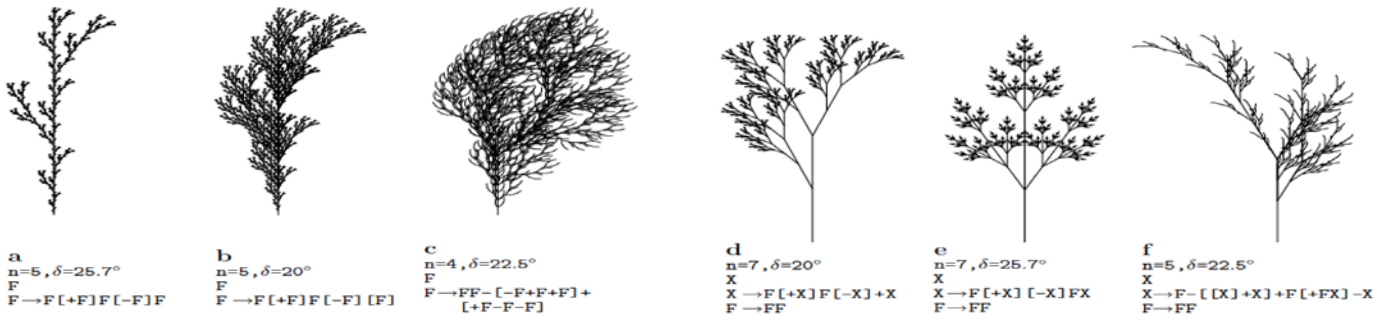
Why Parametric Design?

- Expand design space (from single design to design family)
- Enable design customization
- Constrain design for manufacturability
- Solve inverse problems easier (later)

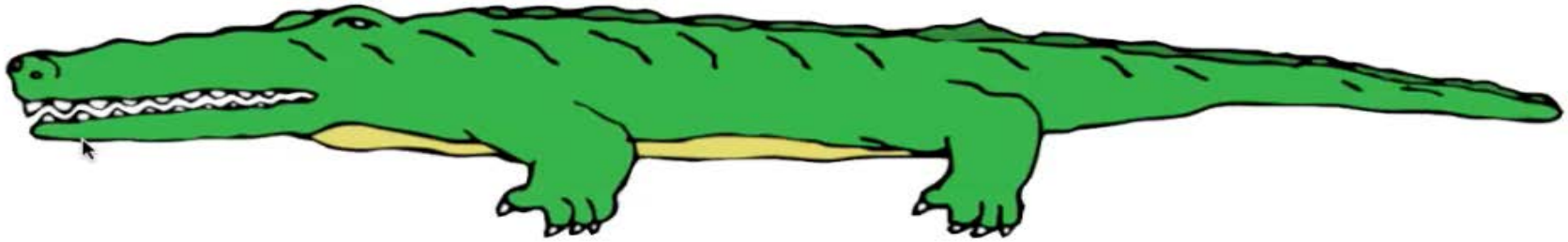
How do we create Parametric Designs?

How do we create Parametric Designs?

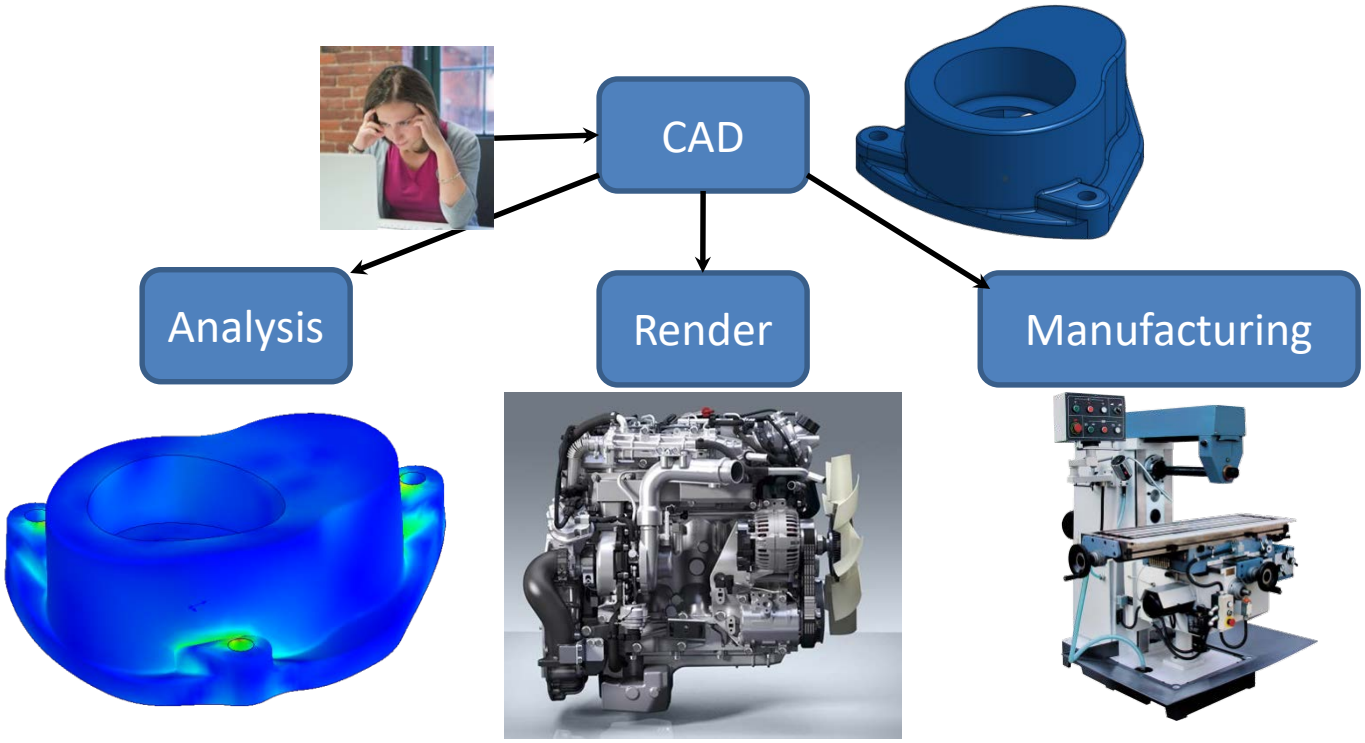
- CAD Systems
- Procedural Modeling



- Deformation Methods



CAD Systems



Source: Ilya Baran

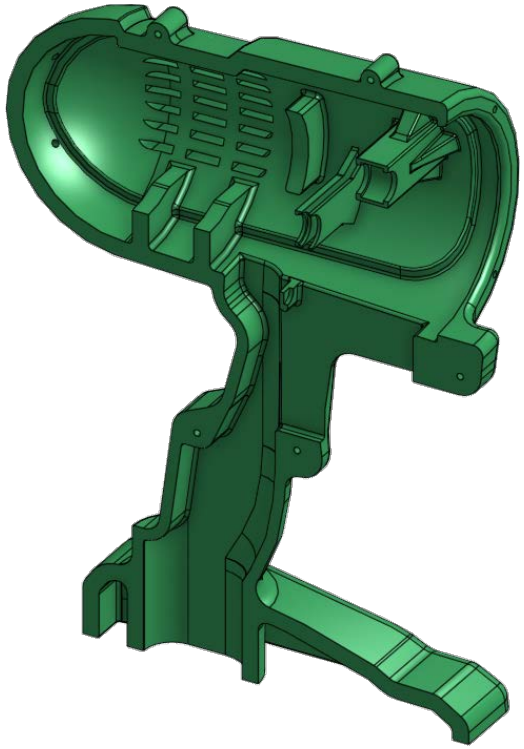
A very brief history of CAD

- 1963—Sketchpad
  - First interactive graphics
- 1970s – 2D
- 1980s – 3D
- 1990s – Parametric CAD and commoditization

Source: Ilya Baran

Parametric feature-based modeling

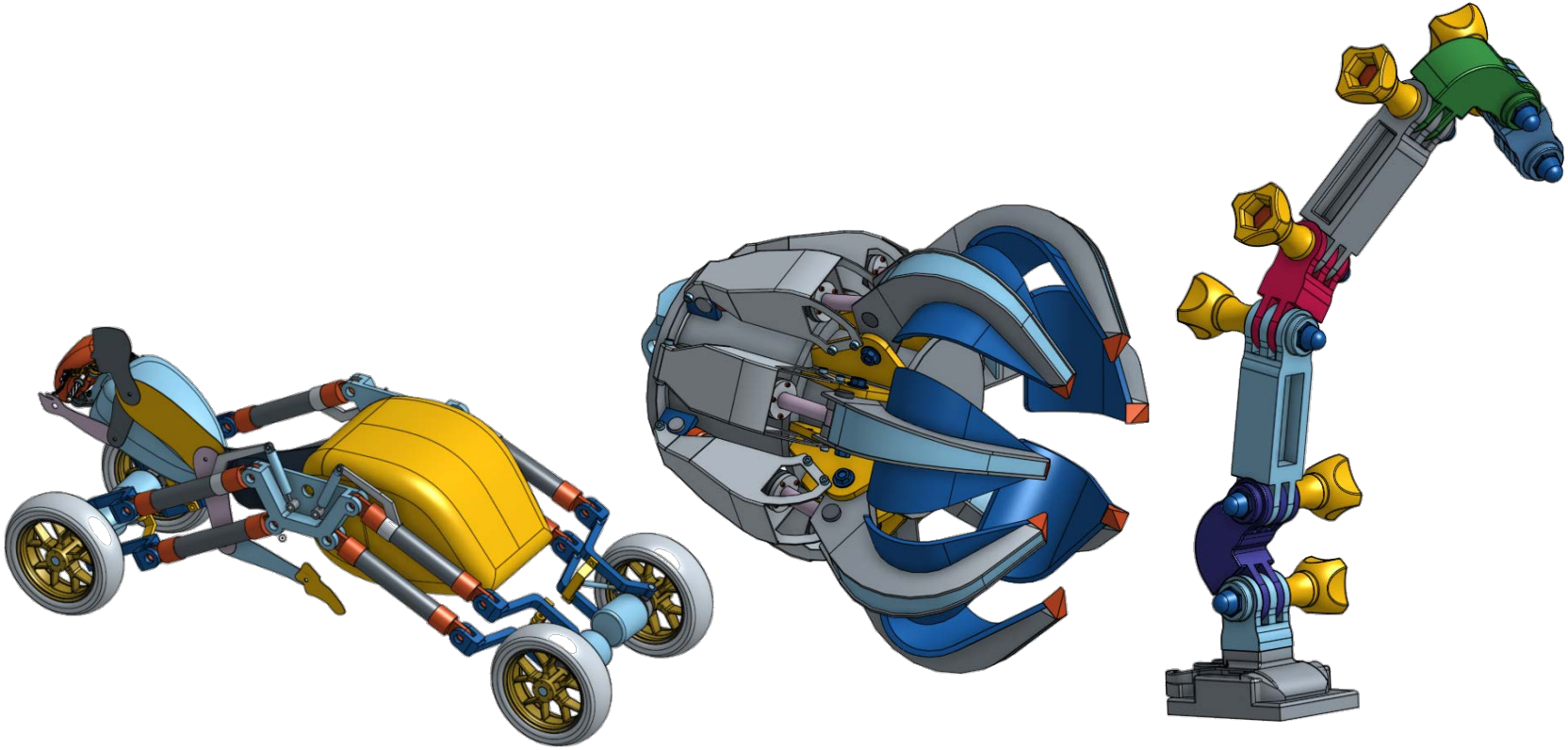
- Theory: easily modify designs and create variations
- Recipe for creating the model
- Start with sketches
- Apply “features”



Source: Ilya Baran

Assemblies

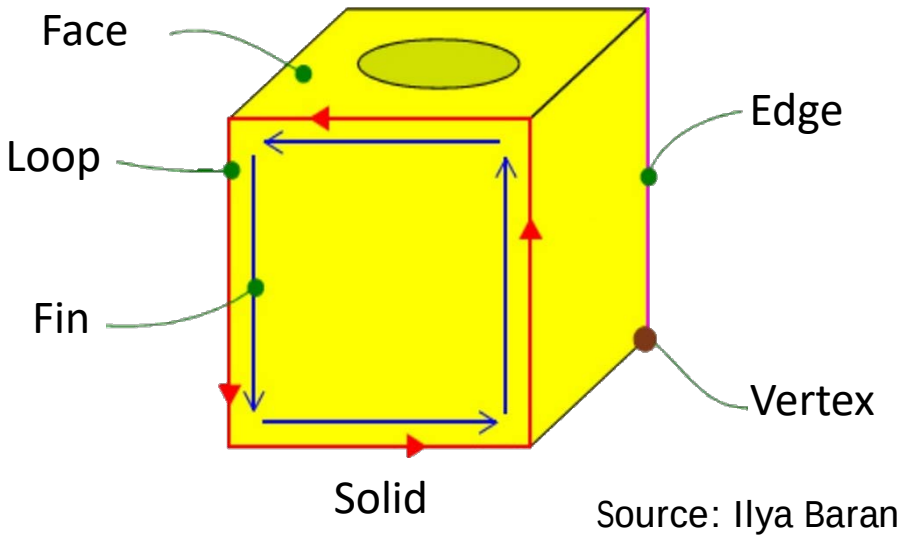
- Hierarchical
- Parts combined via “mates”



Source: Ilya Baran

Boundary Representation (B-REP)

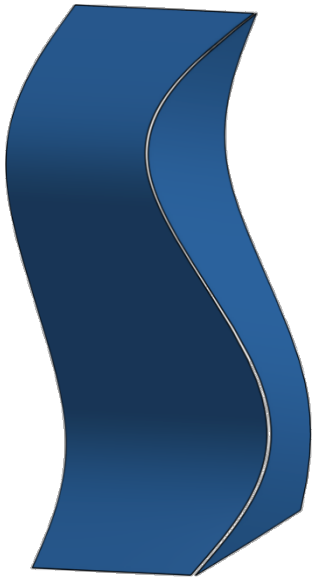
- Topology
  - Graph of edges, vertices, faces, etc.
- Geometry
  - Parametric curves and surfaces
  - “Analytic” and splines



Source: Ilya Baran

Geometry Kernels

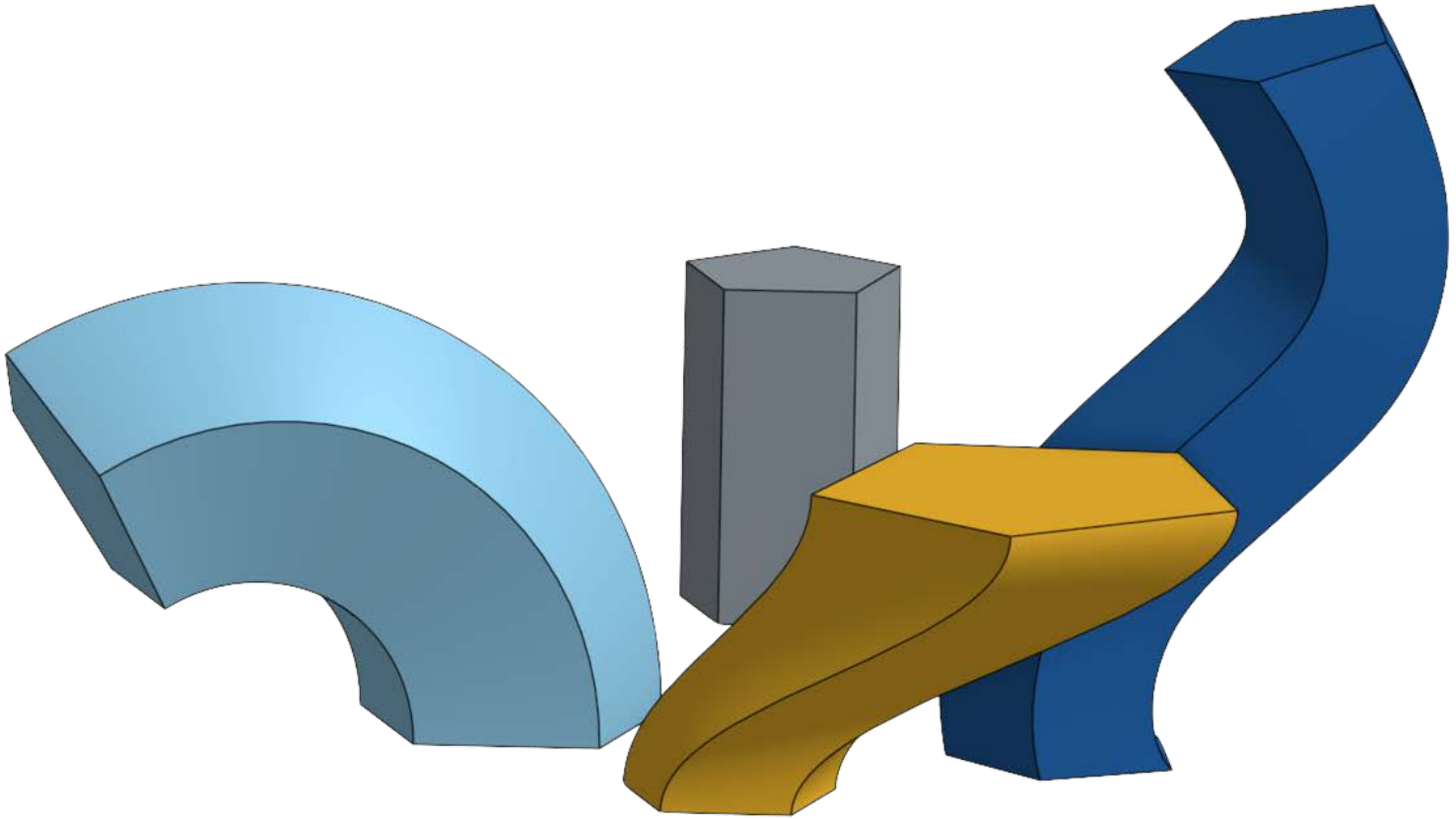
- Standalone: ACIS and Parasolid
- Both descended from Cambridge, UK
- Others not nearly as good
- Key differentiation
  - Robustness
  - Ability to work with bad data
  - Performance



Source: Ilya Baran

Primitive Operations

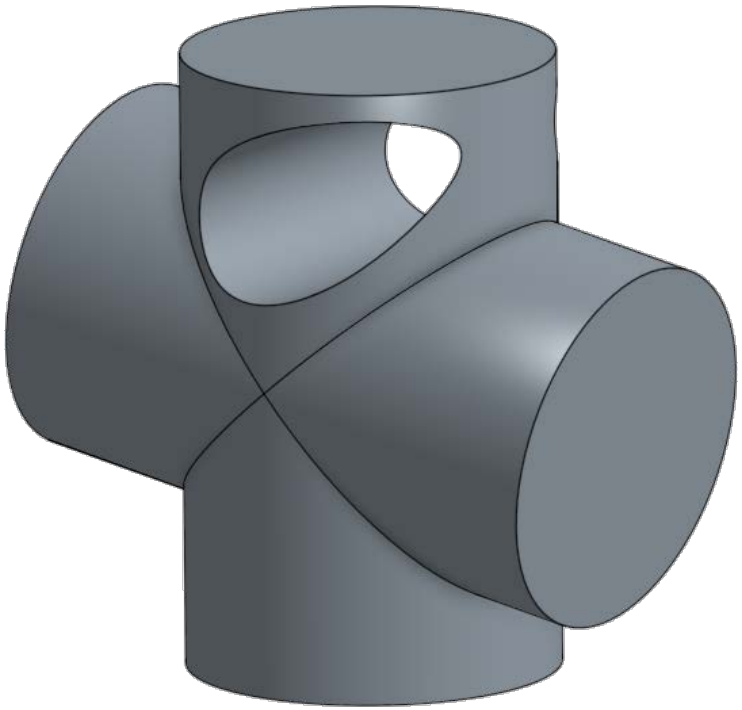
- Sweeping



Source: Ilya Baran

Primitive Operations

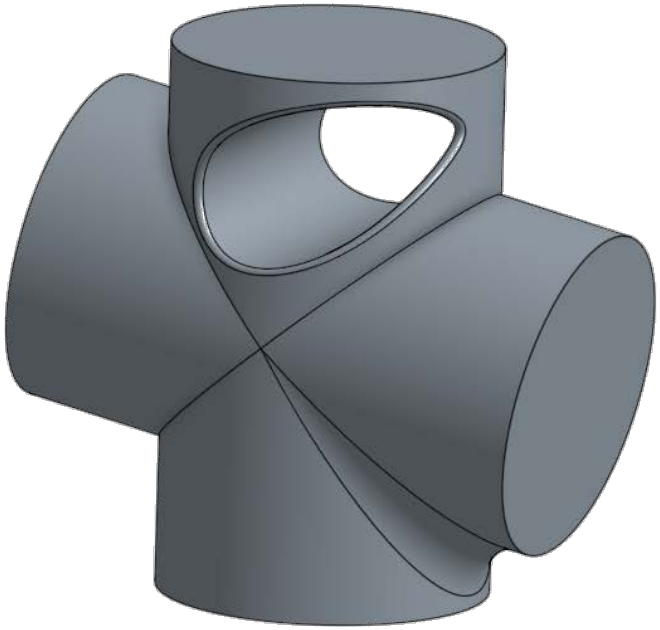
- Sweeping
- Booleans

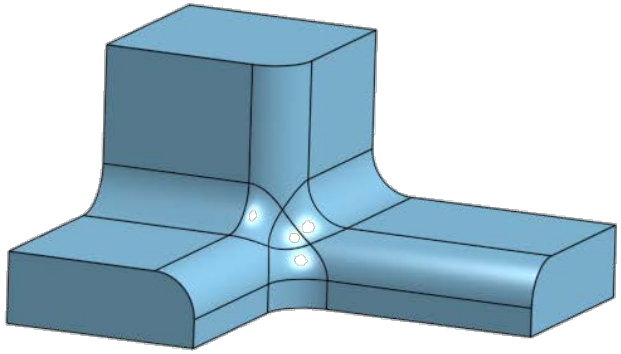


Source: Ilya Baran

Primitive Operations

- Sweeping
- Booleans
- Fillet

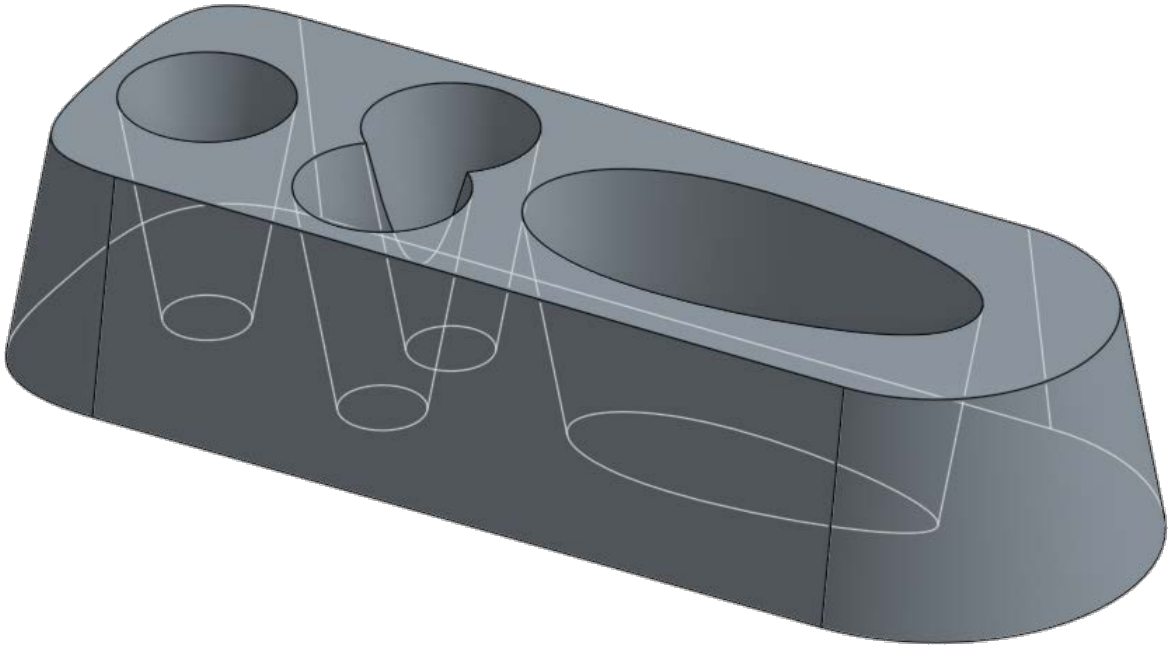




Source: Ilya Baran

Primitive Operations

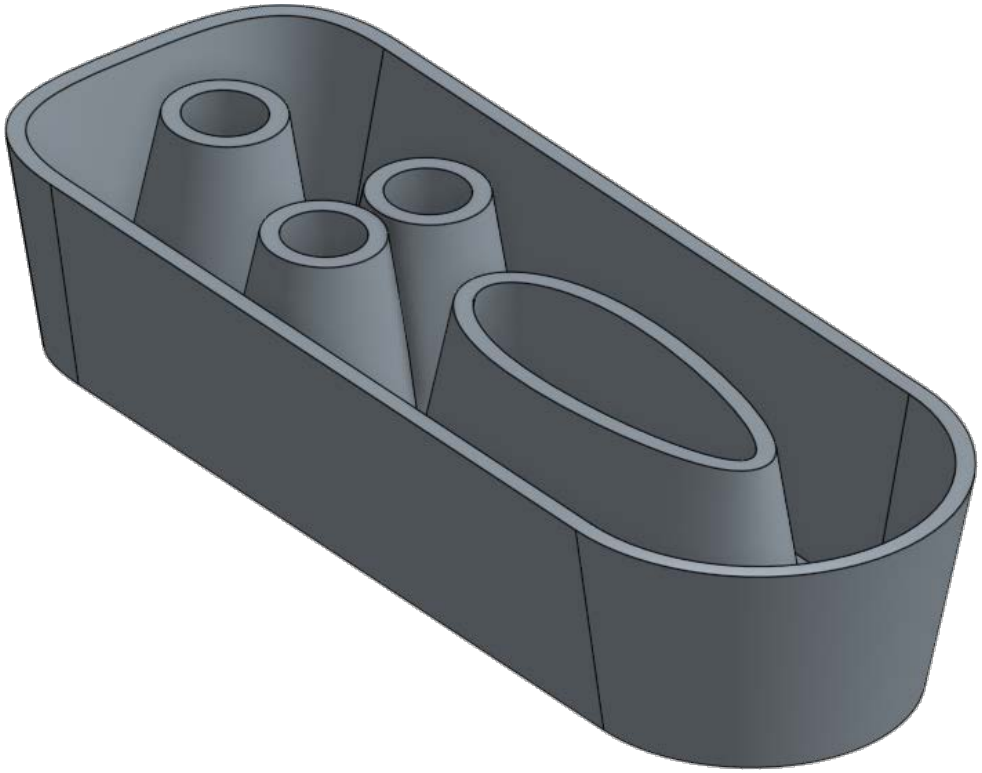
- Sweeping
- Booleans
- Fillet
- Draft



Source: Ilya Baran

Primitive Operations

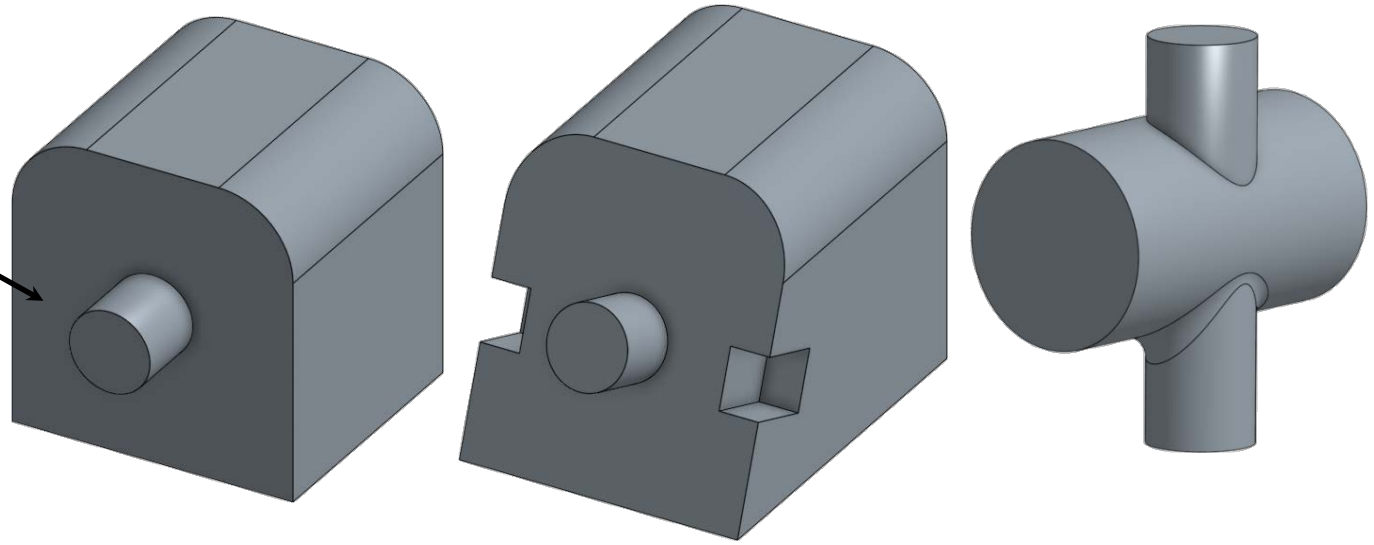
- Sweeping
- Booleans
- Fillet
- Draft
- Offset



Source: Ilya Baran

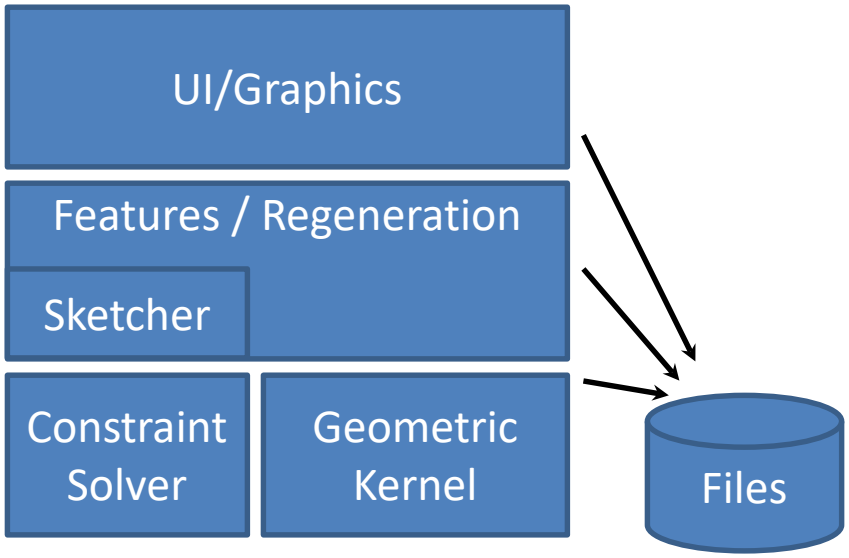
Referencing / Identity

On which face is the protrusion?



Source: Ilya Baran

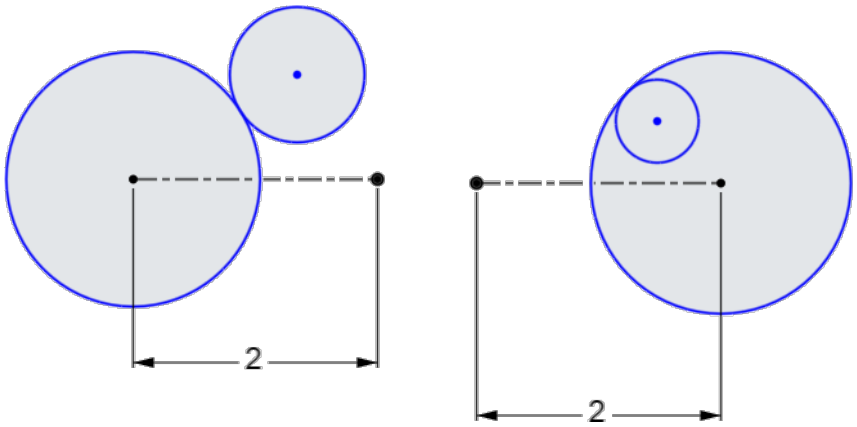
CAD Architecture



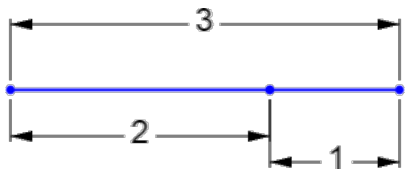
Source: Ilya Baran

Geometric Constraint Solving

- Harder than “just” nonlinear equations:
  - Underdefined
  - Overspecified
  - Disconnected solutions



- Users expect “minimal” change



Source: Ilya Baran

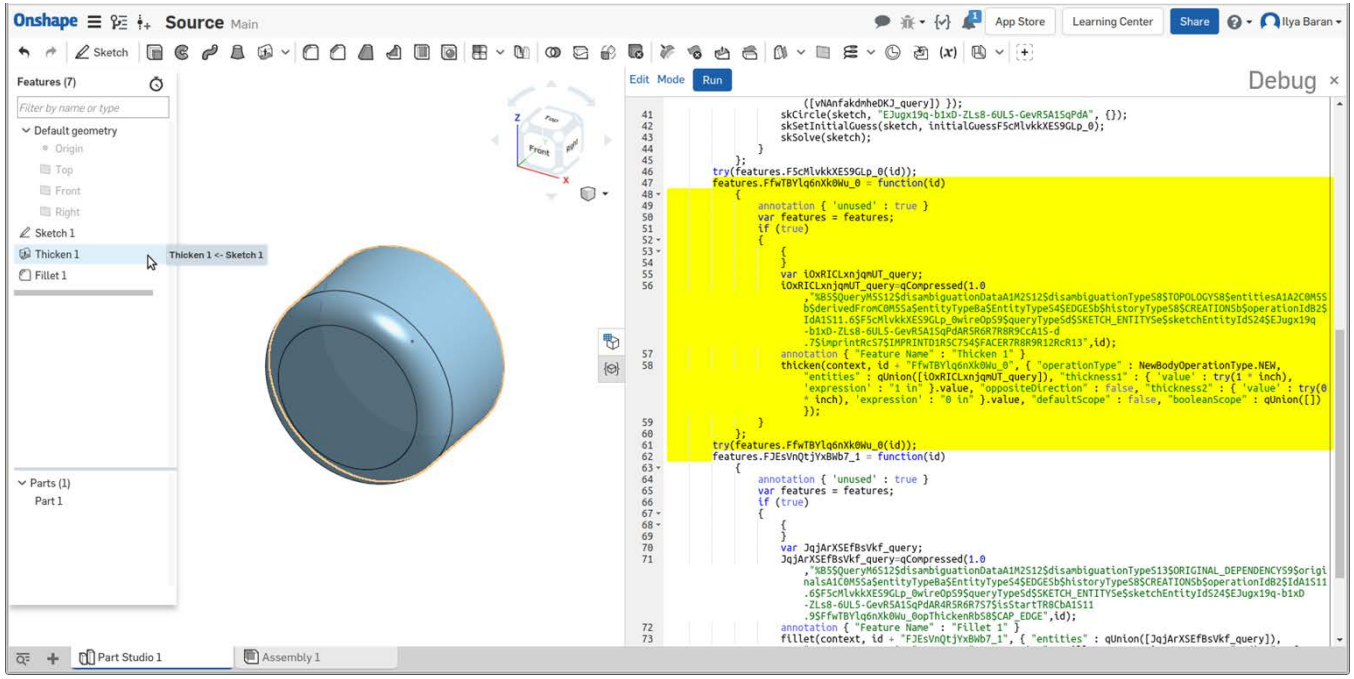
CAD has many parallels to programming

Programming	⟷	**CAD**
Program	⟷	Parametric Model
Function	⟷	Feature, Part Studio
Loop	⟷	Pattern
Version Control	⟷	Product Data Management (PDM)
Code diff	⟷	Model compare

Source: Ilya Baran

Programming in CAD (Onshape)

- A Part Studio in Onshape is a function



Source: Ilya Baran

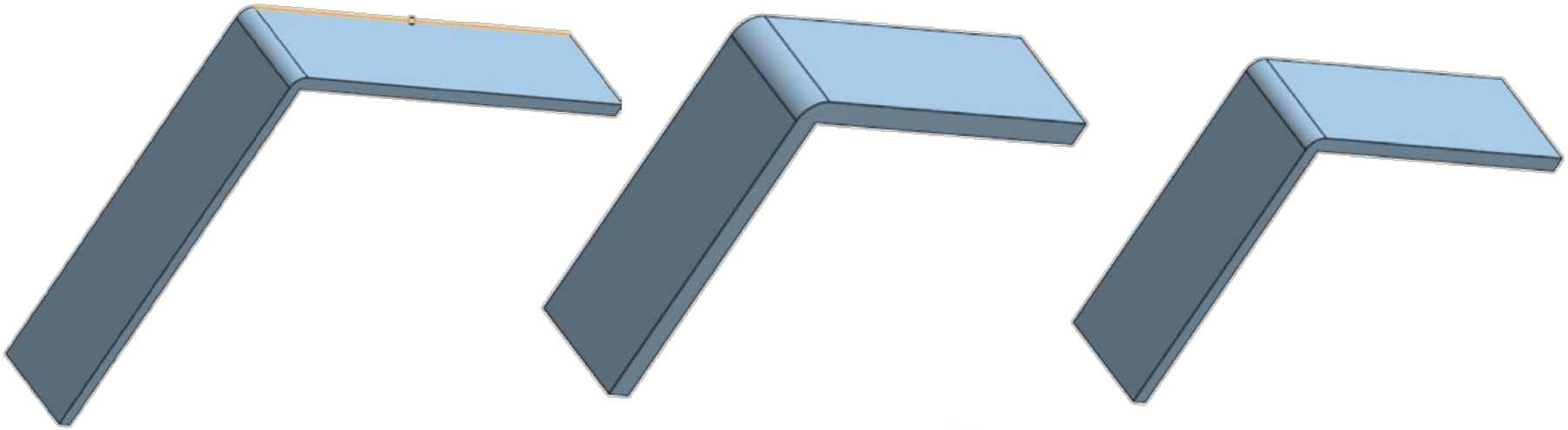
FeatureScript (Onshape)

- New language for writing CAD features
- All Onshape features are written in FeatureScript and are open source
- Users can make their own features
- www.onshape.com/featurescript

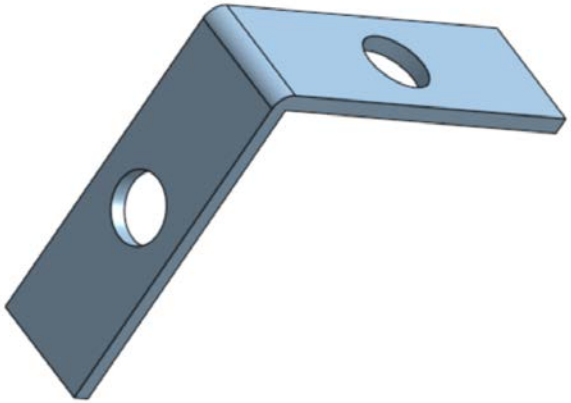
Source: Ilya Baran

Parametric Model in OnShape (Lab)

- Create a model with a small number of parameters

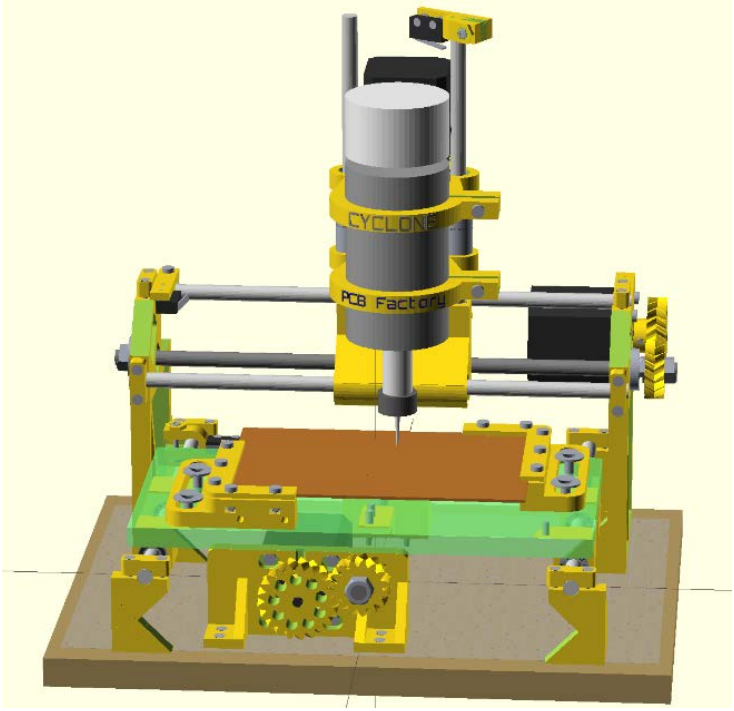


- Design custom features



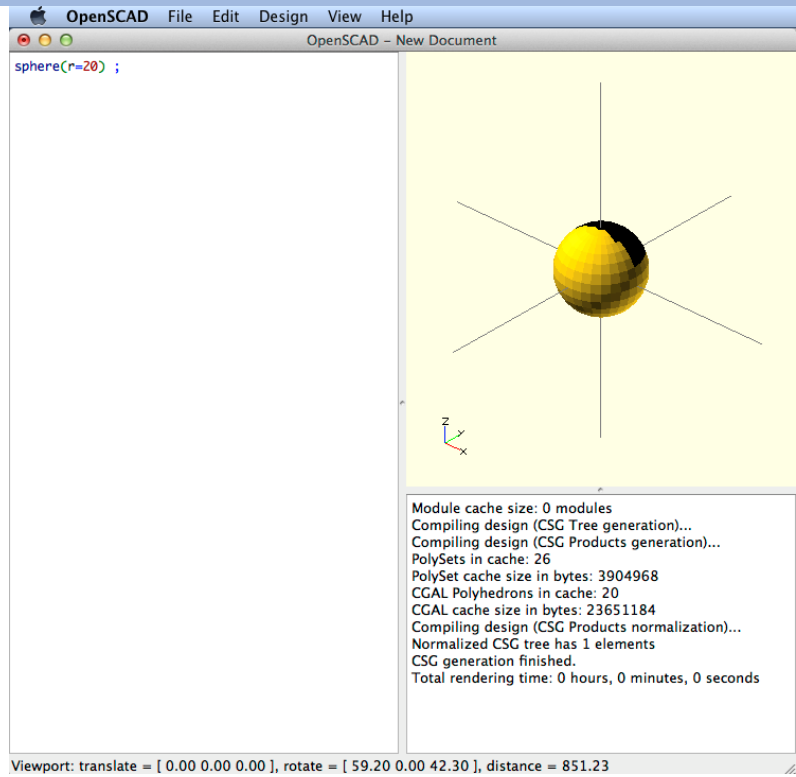
OpenSCAD

- Software for creating solid 3D CAD models
- Not an interactive modeler
  - Very basic UI
- A 3D-compiler
  - Geometry written as a script
  - Executed using CGAL/OpenCSG
  - Rendered with OpenGL
- Available for Linux/UNIX, Windows, Mac OS X
  - http://www.openscad.org



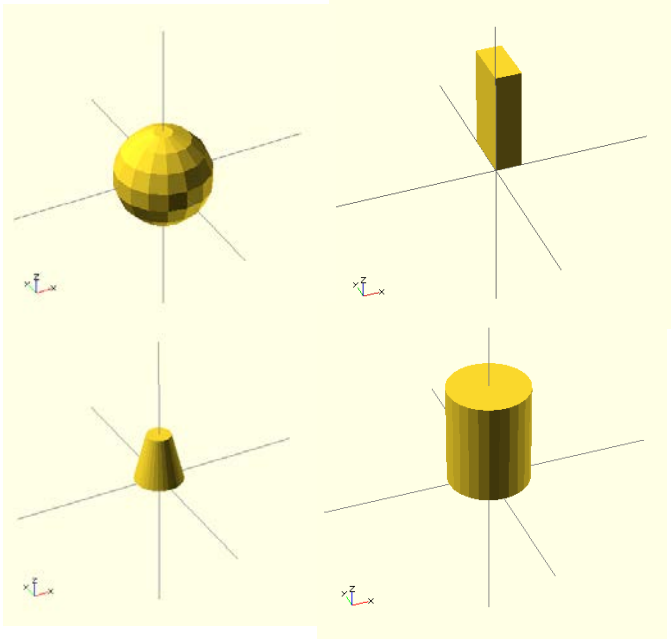
OpenSCAD

- Interface
  - 3 panels
    - Script
    - View
    - Info
- Compile (F5)
  - Design->Compile
- Show Axes (Ctrl+2)



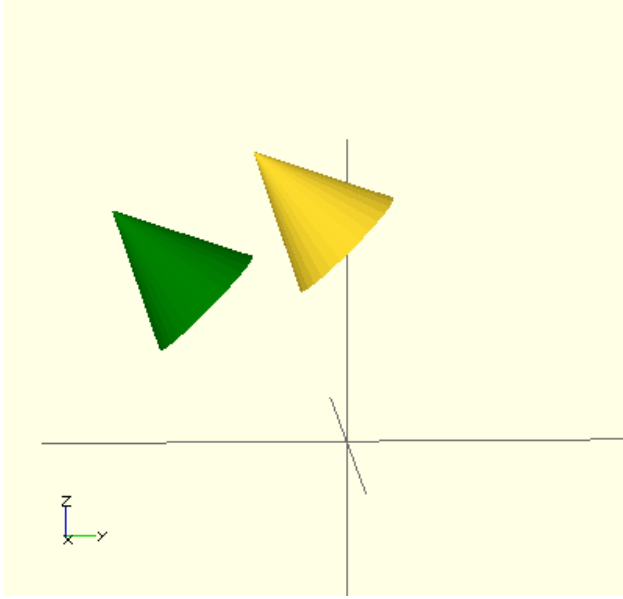
3D Primitives

- Sphere
  - `sphere(5); sphere(r=5);`
- Cube
  - `cube(5);`
  - `cube([4,8,16]);`
- Cylinder
  - `cylinder(20,10,5);`
    `cylinder(h = 20, r1 = 10,`
    `r2 = 5);`
  - `cylinder(h=20,r=10);`



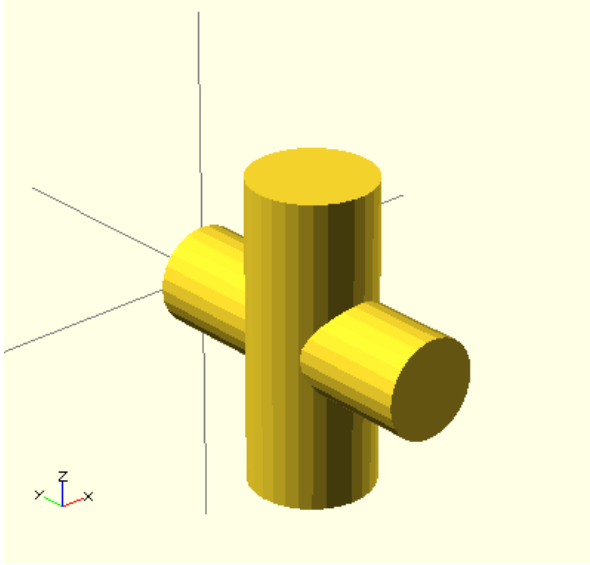
Transformations

- Translate
  - e.g., `translate([10,0,0]) sphere(5); // translate along x axis`
- Rotate
- Scale
- Order dependent
  - `translate([0,0,10]) rotate([45,0,0]) cylinder([20,10,0]);`
  - `Color(“green”) rotate([45,0,0]) translate([0,0,10]) cylinder([20,10,0]);`



CSG

- Union
- Intersection
- Difference
- Example:

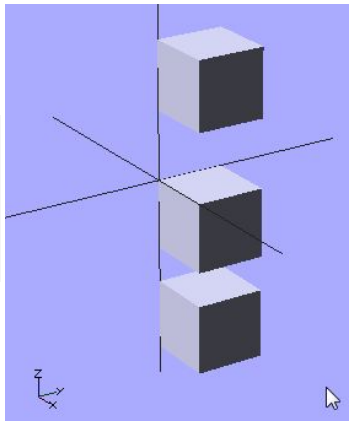


```
union()
 {
 translate([0,-25,-25]) cylinder(50,10,10);
 rotate([90,0,0]) cylinder(50,8,8);
 }
```

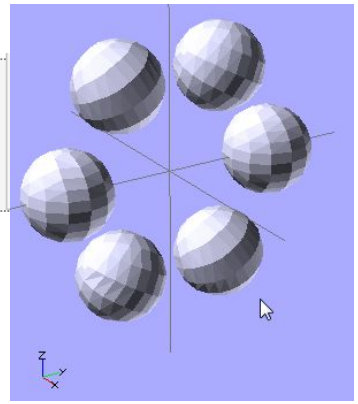
Loops

```
for (loop_variable_name = range or vector) {
    …..
      }
```

```
for ( z = [-1, 1, -2.5]) {
 translate( [0, 0, z] )
    cube(size = 1, center = false);
}
```

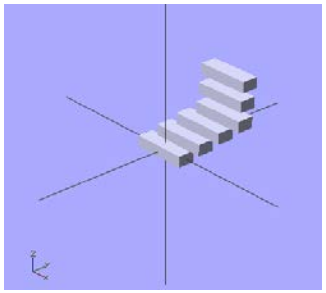


```
 for ( i = [0:5] ) {
  rotate( i*360/6, [1, 0, 0])
   translate( [0, 10, 0] ) sphere(r = 1);
 }
```

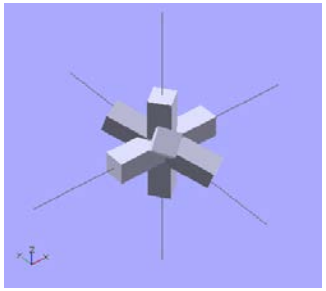


Loops

```
for(i = [ [ 0,  0,  0],
          [10, 12, 10],
          [20, 24, 20],
          [30, 36, 30],
          [20, 48, 40],
          [10, 60, 50] ])
{
    translate(i)
    cube([50, 15, 10], center = true);
}
```



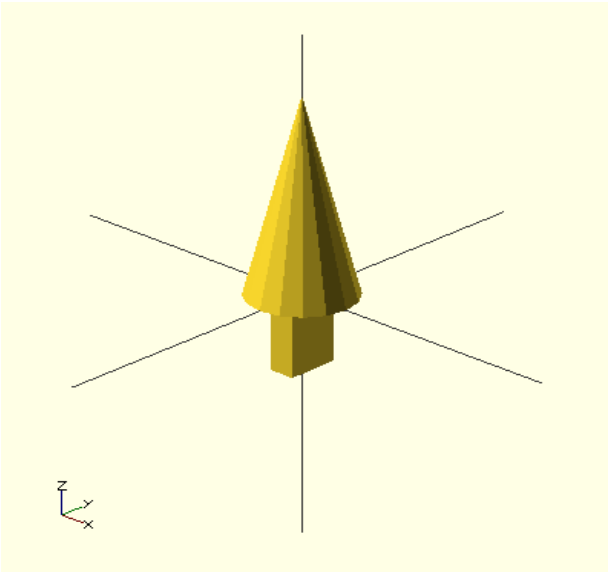
```
for(i = [ [  0,  0,   0],
          [ 10, 20, 300],
          [200, 40,  57],
          [ 20, 88,  57] ])
{
    rotate(i)
    cube([100, 20, 20], center = true);
}
```



Module

- Procedures/Functions

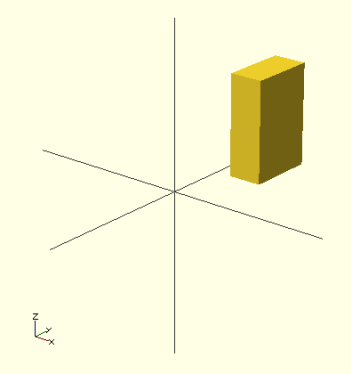
```
module leaves() { cylinder(20,5,0); }
module box() { cube([5,10,15]); }

module tree() {
  leaves();
  scale([0.5,0.5,0.5]) translate([-2.5,-5,-15]) box();
   }

tree();
```



Module

- Parameters

```
module box(w,l,h,tx,ty,tz){
        translate([tx,ty,tz])
cube([w,l,h]);
}
box(5,10,15,10,0,5);
```

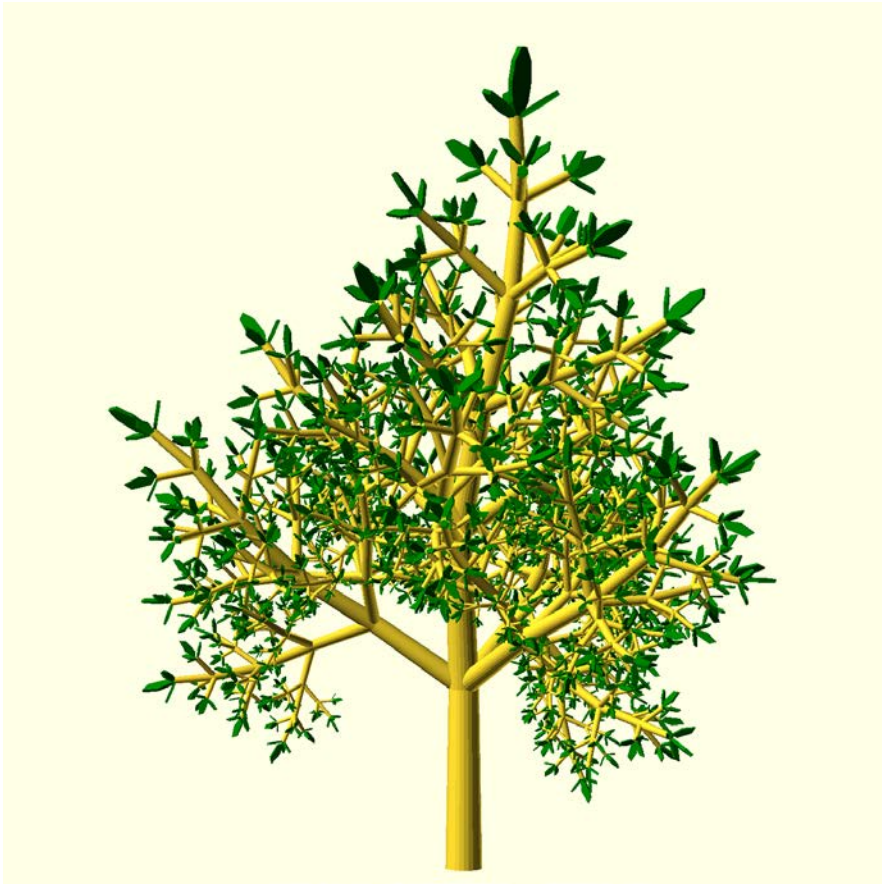


- Default values

```
module box2(w=5,l=10,h=20){
  echo("w=", w, " l=", l, " h=", h);
  cube([w,l,h]);
}
box2();
```

Parametric Model in OpenSCAD

```
module simple_tree(size, dna, n) {
    if (n > 0) {
        // trunk
        cylinder(r1=size/10, r2=size/12, h=size, $fn=24);
        // branches
        translate([0,0,size])
            for(bd = dna) {
                angx = bd[0];
                angz = bd[1];
                scal = bd[2];
                    rotate([angx,0,angz])
                        simple_tree(scal*size, dna, n-1);
            }
    }
    else // leaves
        color("green")
        scale([1,1,3])
            translate([0,0,size/6])
                rotate([90,0,0])
                    cylinder(r=size/6,h=size/10);
}
// dna is a list of branching data bd of the tree:
//      bd[0] - inclination of the branch
//      bd[1] - Z rotation angle of the branch
//      bd[2] - relative scale of the branch
dna = [ [12,  80, 0.85], [55,    0, 0.6],
        [62, 125, 0.6], [57, -125, 0.6] ];
simple_tree(50, dna, 6);
```



Graphics vs. CAD

Modeling for graphics

- Movies, games, images
- Complexity from shaders, tricks
- Meshes, subdivision
- Research in touch with industry

CAD

- Manufacturing, analysis, drawings, images
- Complexity is modeled
- Photorealism can hurt
- B-rep/NURBS, subdivision
- Research disconnected from reality

Source: Ilya Baran