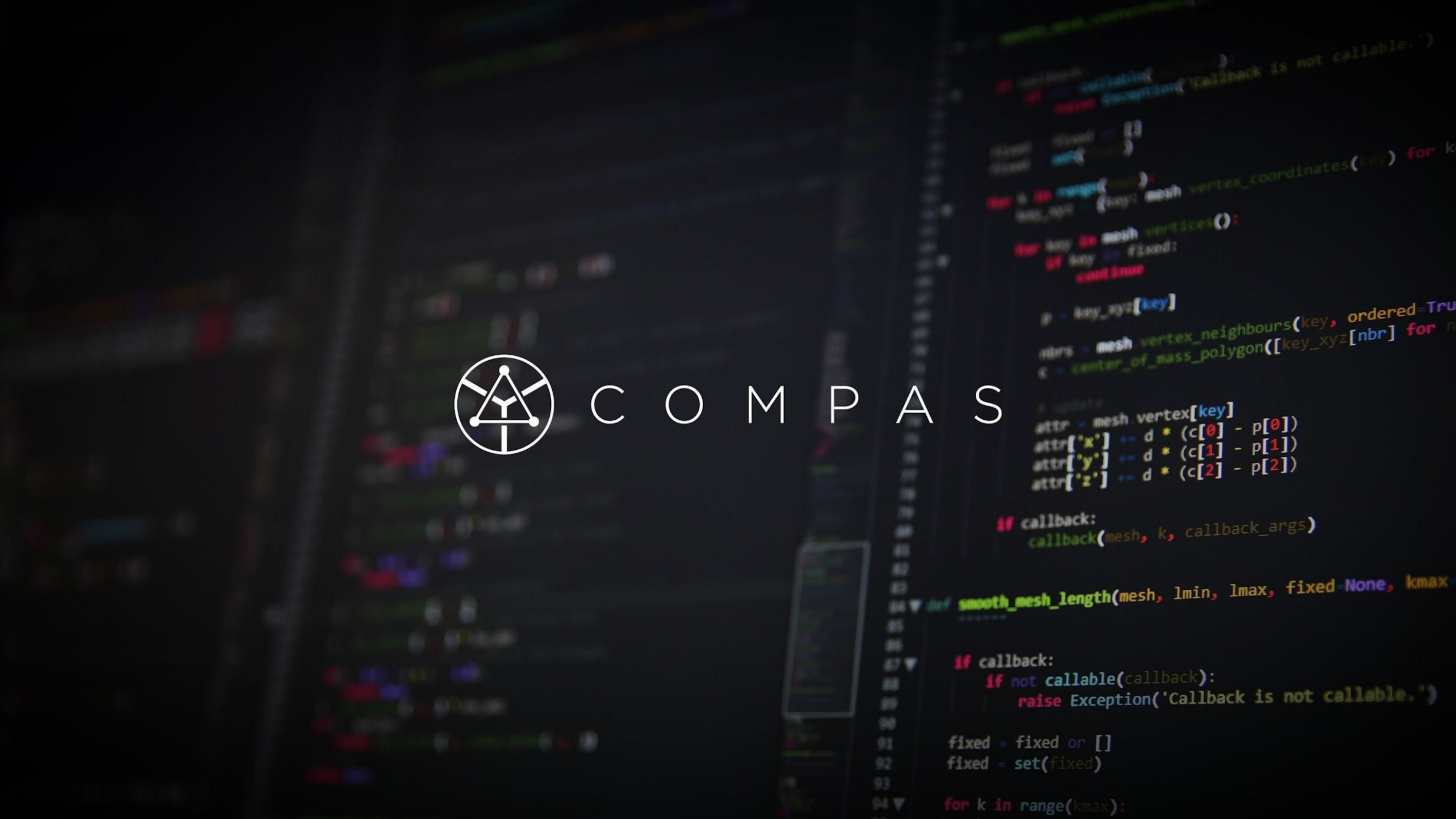
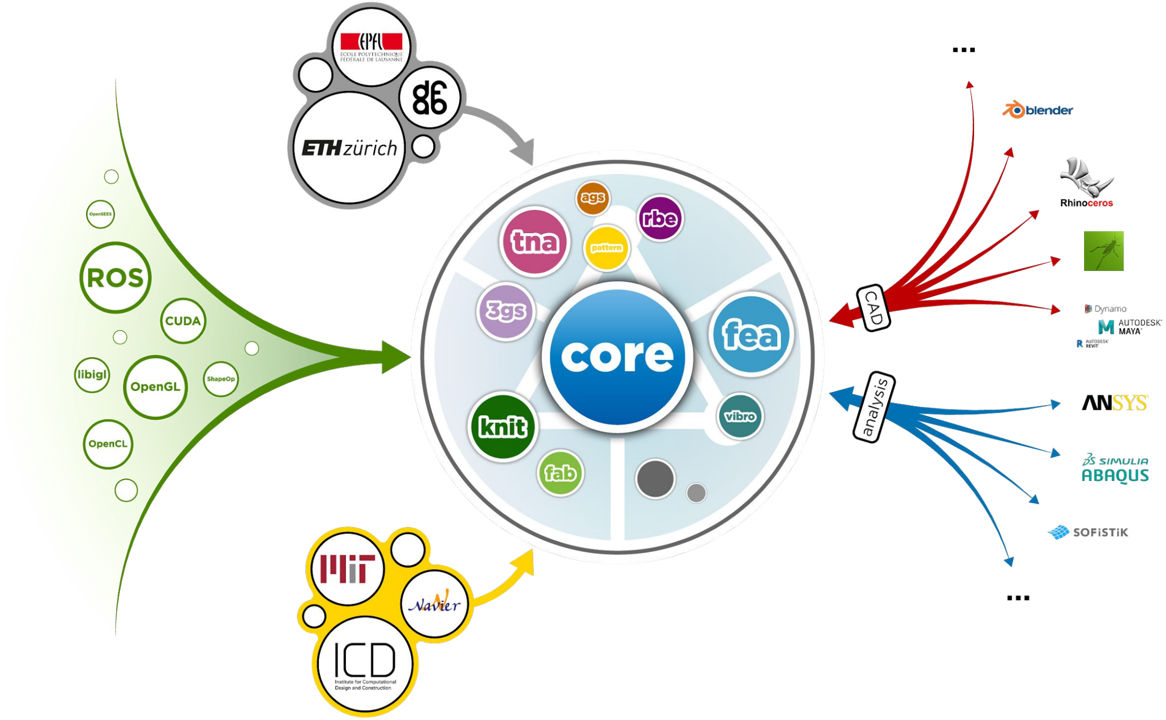
COMPAS


for key in mesh.vertices():
if key in fixed:
continue
p = key_xyz[key]
nbrs = mesh.vertex_neighbours(key, ordered=Tru
c = center_of_mass_polygon([key_xyz[nbr] for
attr = mesh.vertex[key]
attr['x'] += d * (c[0] - p[0])
attr['y'] += d * (c[1] - p[1])
attr['z'] += d * (c[2] - p[2])
if callback:
callback(mesh, k, callback_args)
def smooth_mesh_length(mesh, lmin, lmax, fixed=None, kmax
if callback:
if not callable(callback):
raise Exception('Callback is not callable.')
fixed = fixed or []
fixed = set(fixed)
for k in range(kmax):

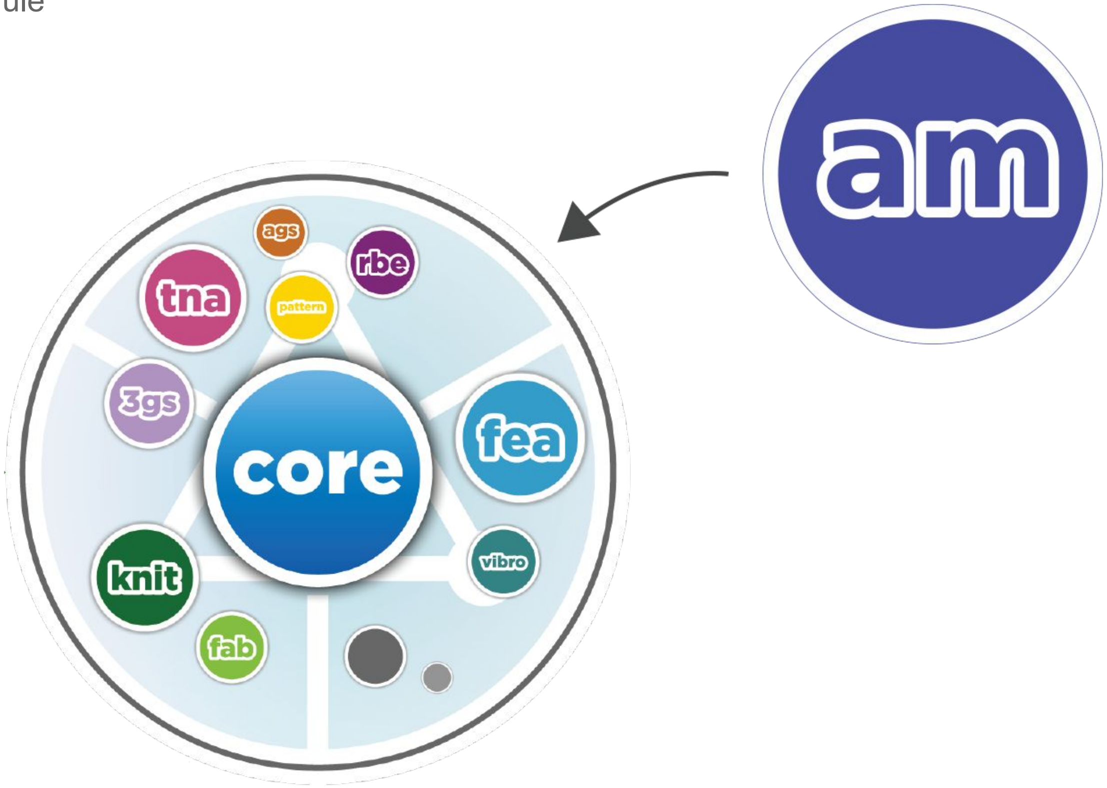
Compas AM

3D Printing module



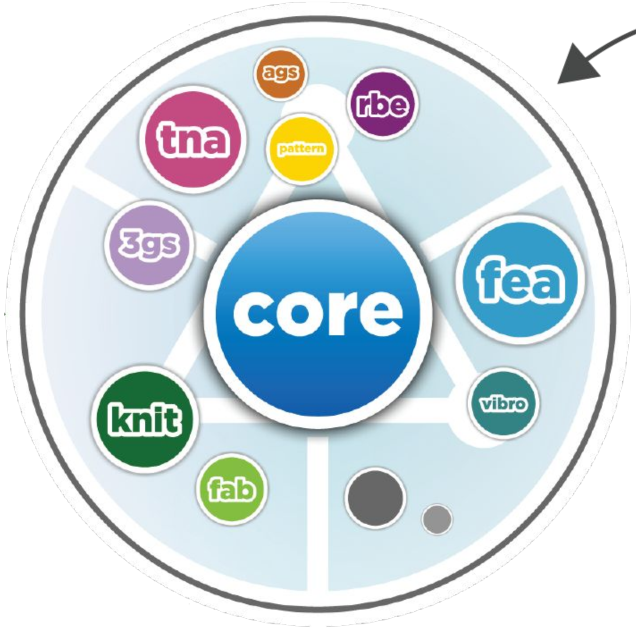
Compas AM

3D Printing module



Compas AM

3D Printing module

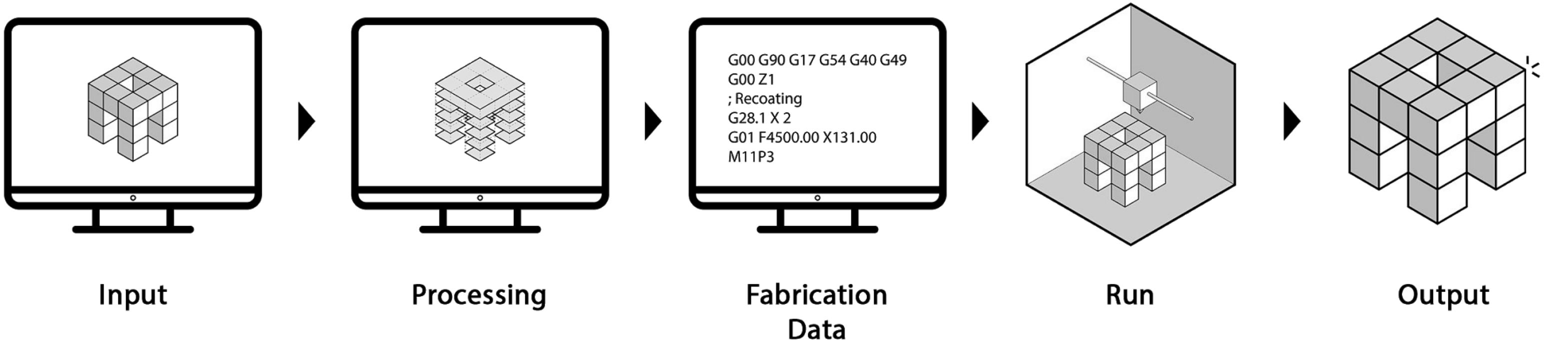


am

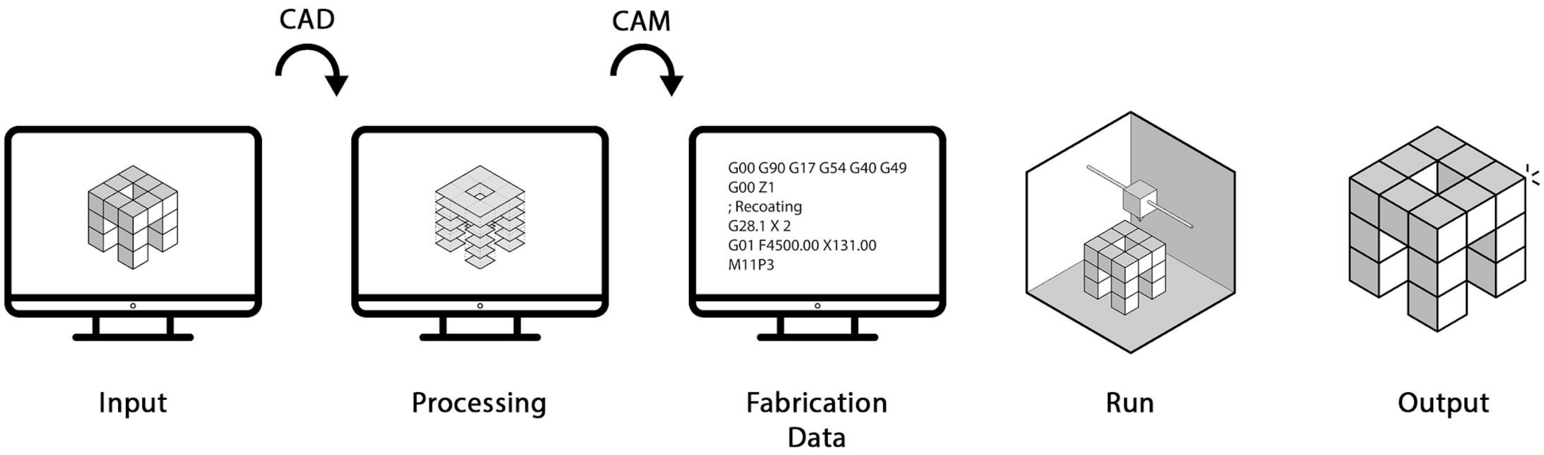
- Basic concepts
- Open discussion
- Demo

Compas AM

Bits and atoms

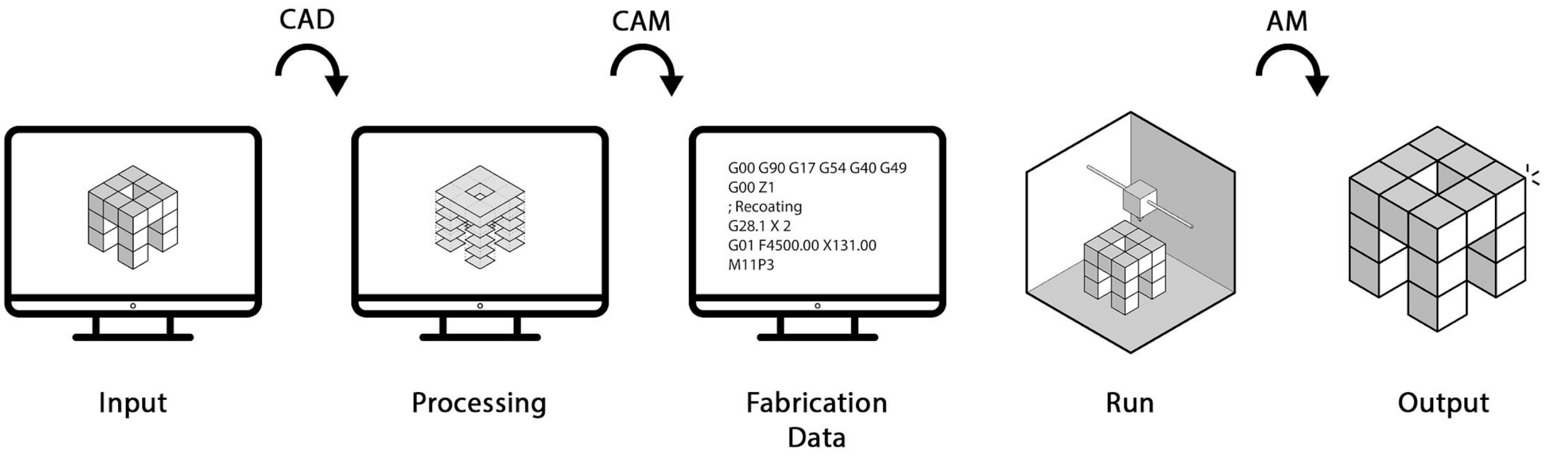


Compas AM
Software



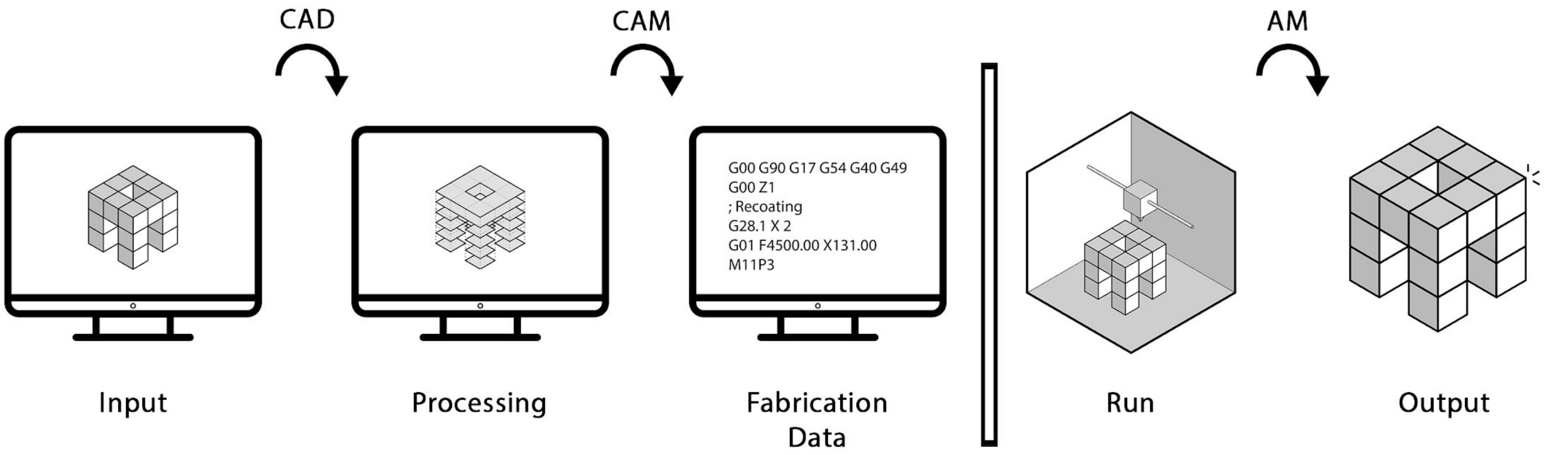
Compas AM

Hardware

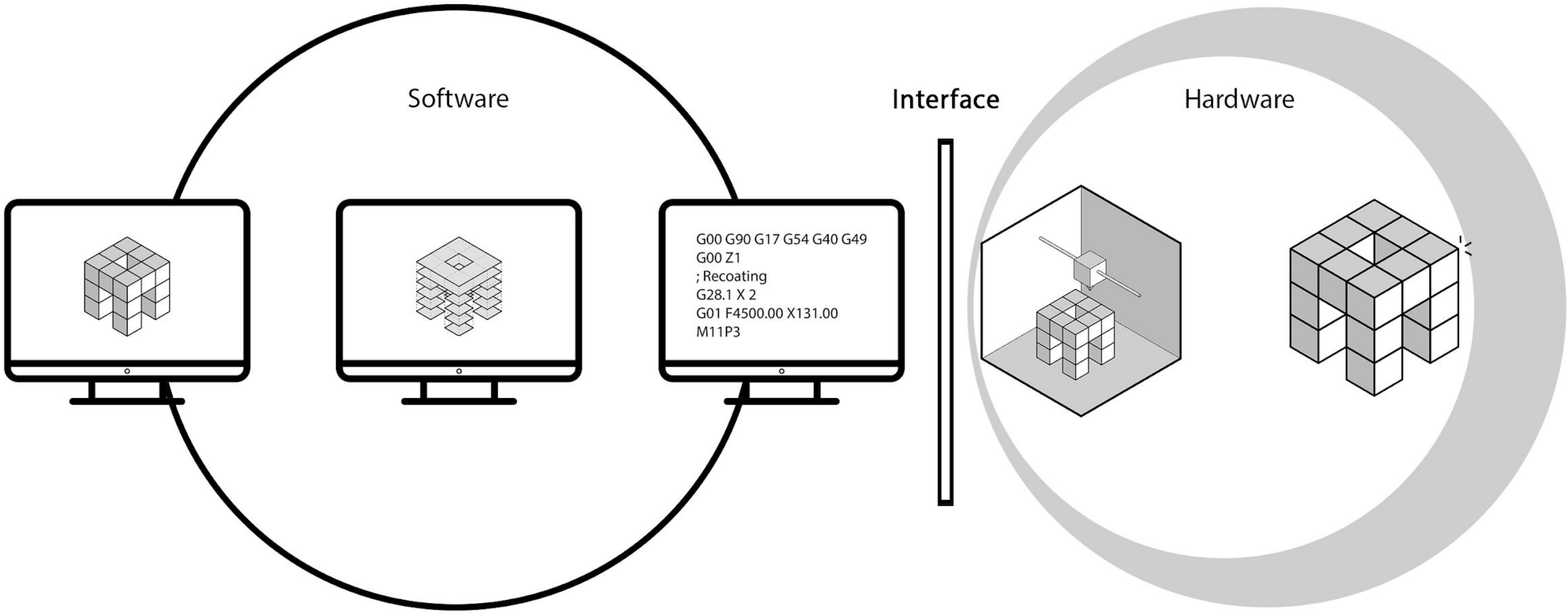


Compas AM

Gap

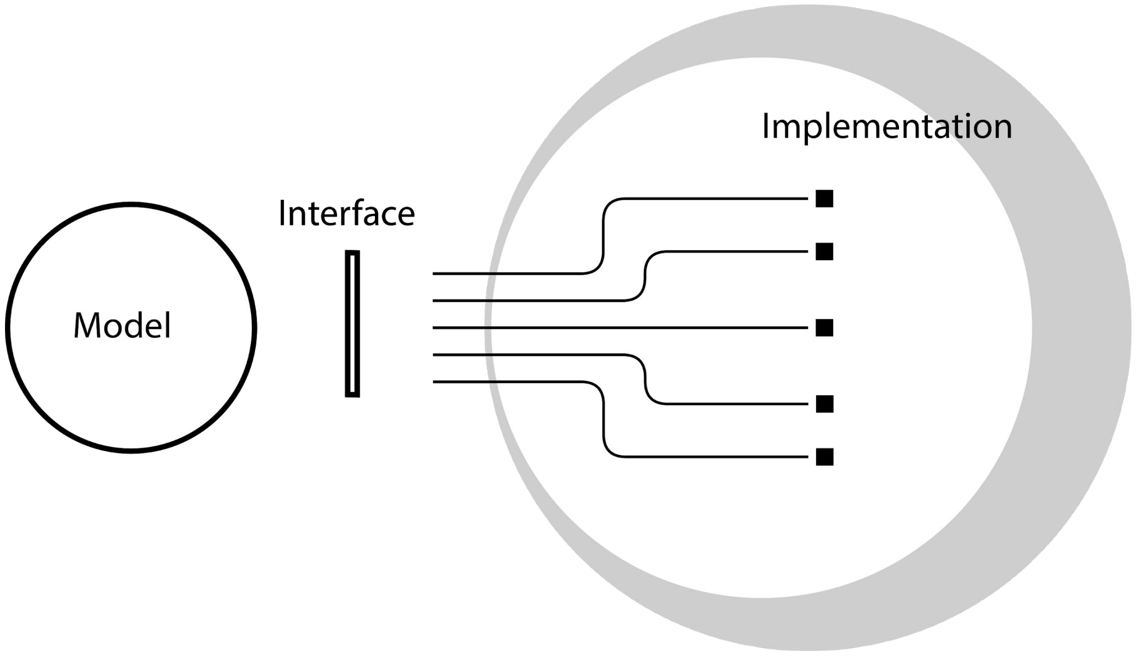


Compas AM
Systems



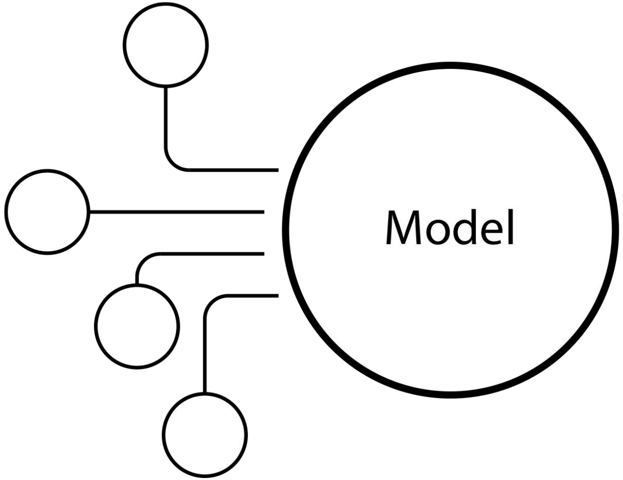
Framework elements

Abstract



Framework elements

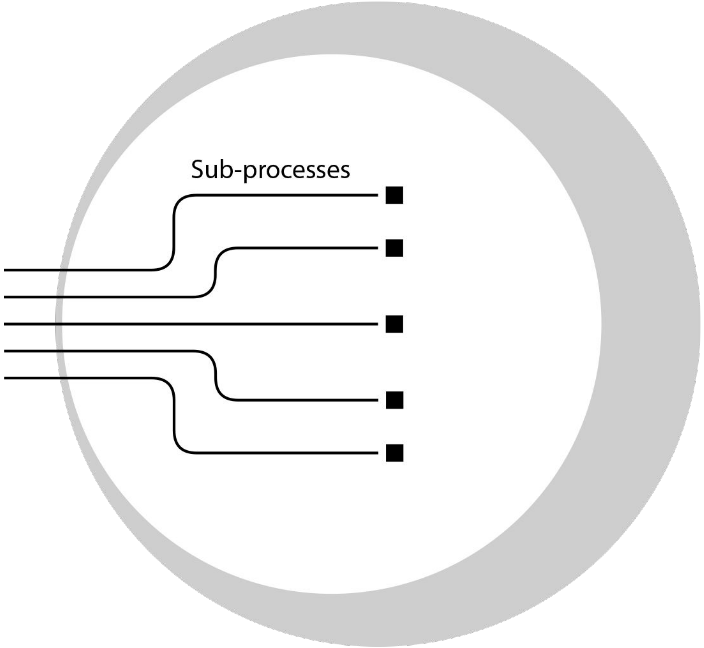
Model



- Contains information gathered from multiple contexts
- Represents the object according to a “code”
- As simple and self contained as possible

Framework elements

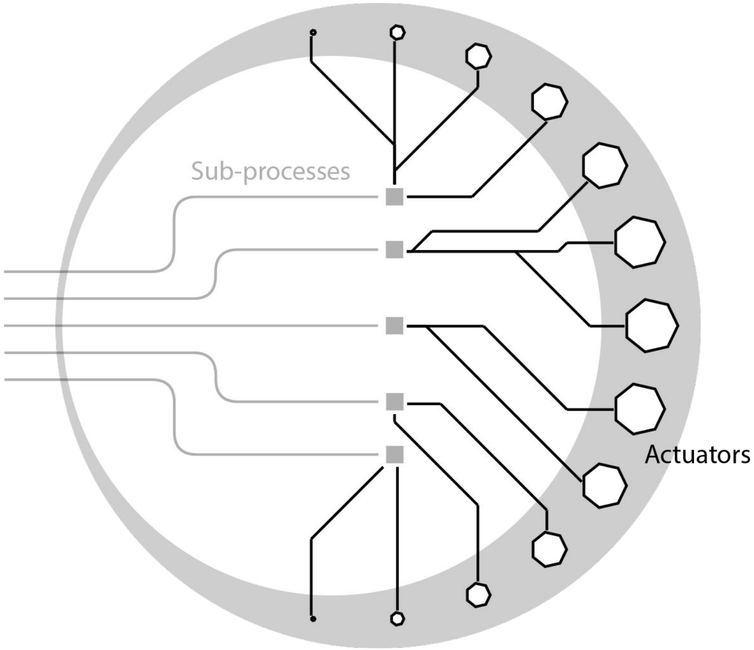
Implementation



- Breaks out the data flow into different branches
- Delivers process-specific instructions
- Reflects hardware configuration

Framework elements

Implementation



- Breaks out the data flow into different branches
- Delivers process-specific instructions
- Reflects hardware configuration

Framework elements

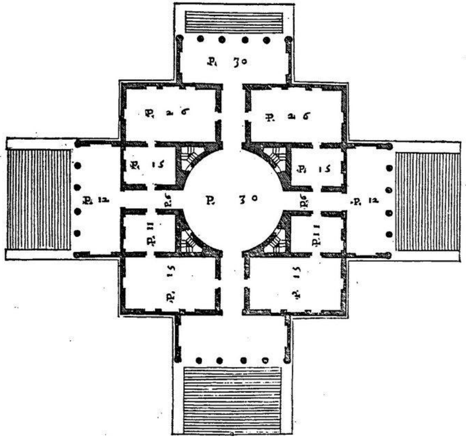
Interface

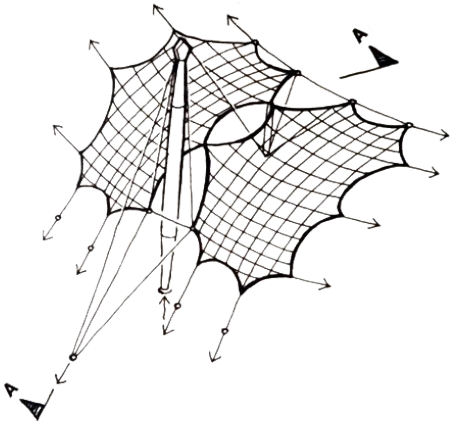
- Knows how to extract and manipulate data from the model

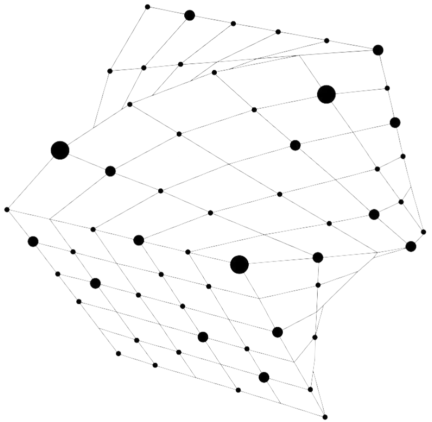
- … and how to convert it into process readable instructions

Framework elements

Patterns

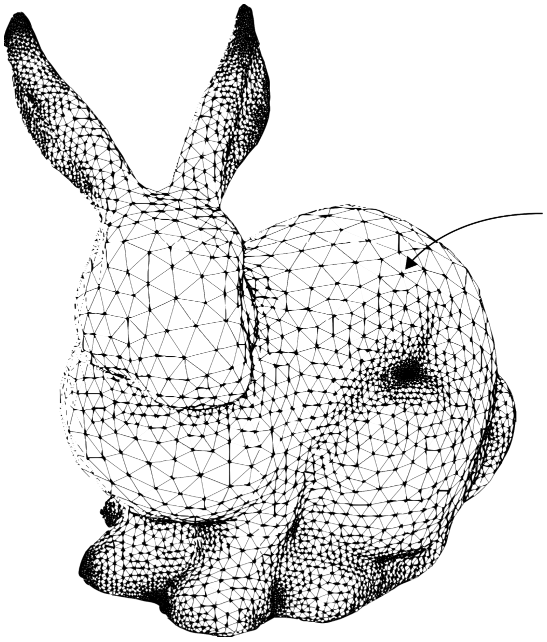






Framework elements

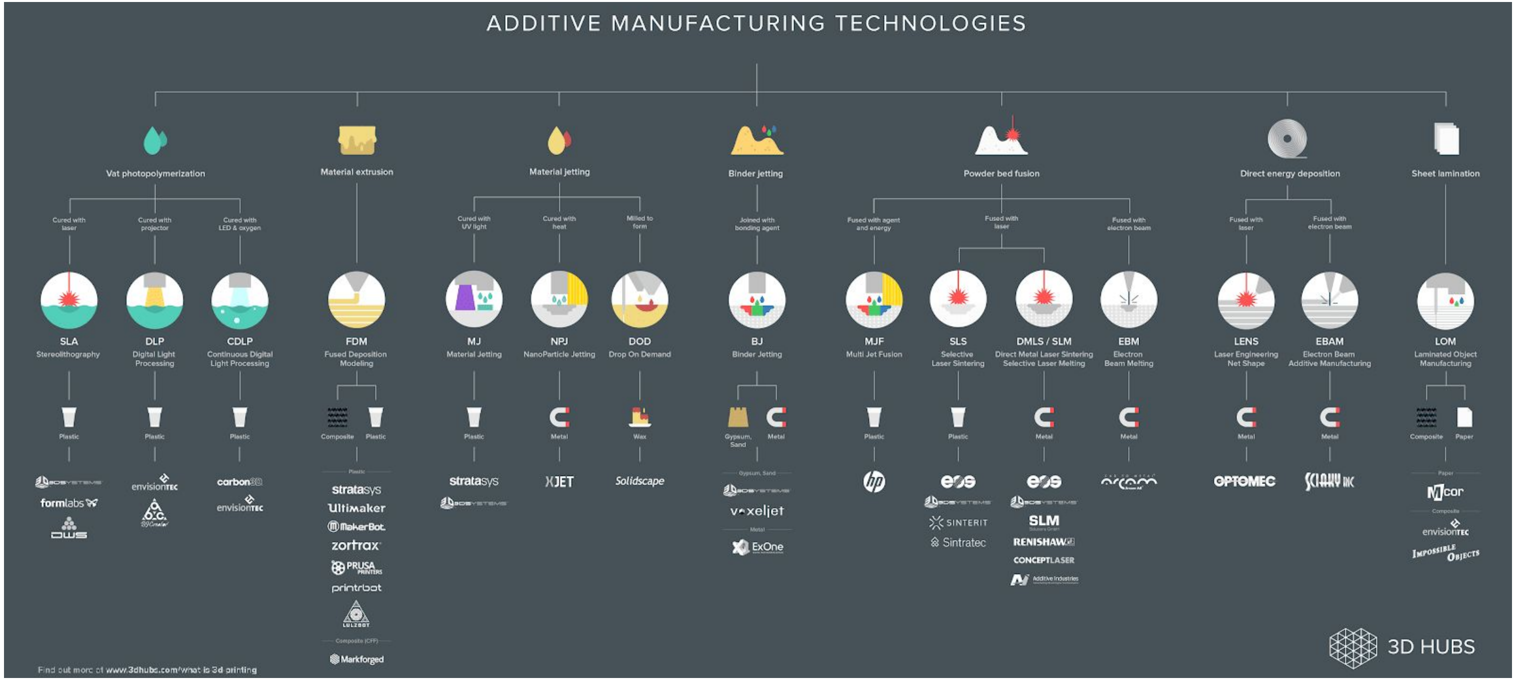
Data processing



n_i {x, y, z, [p]}

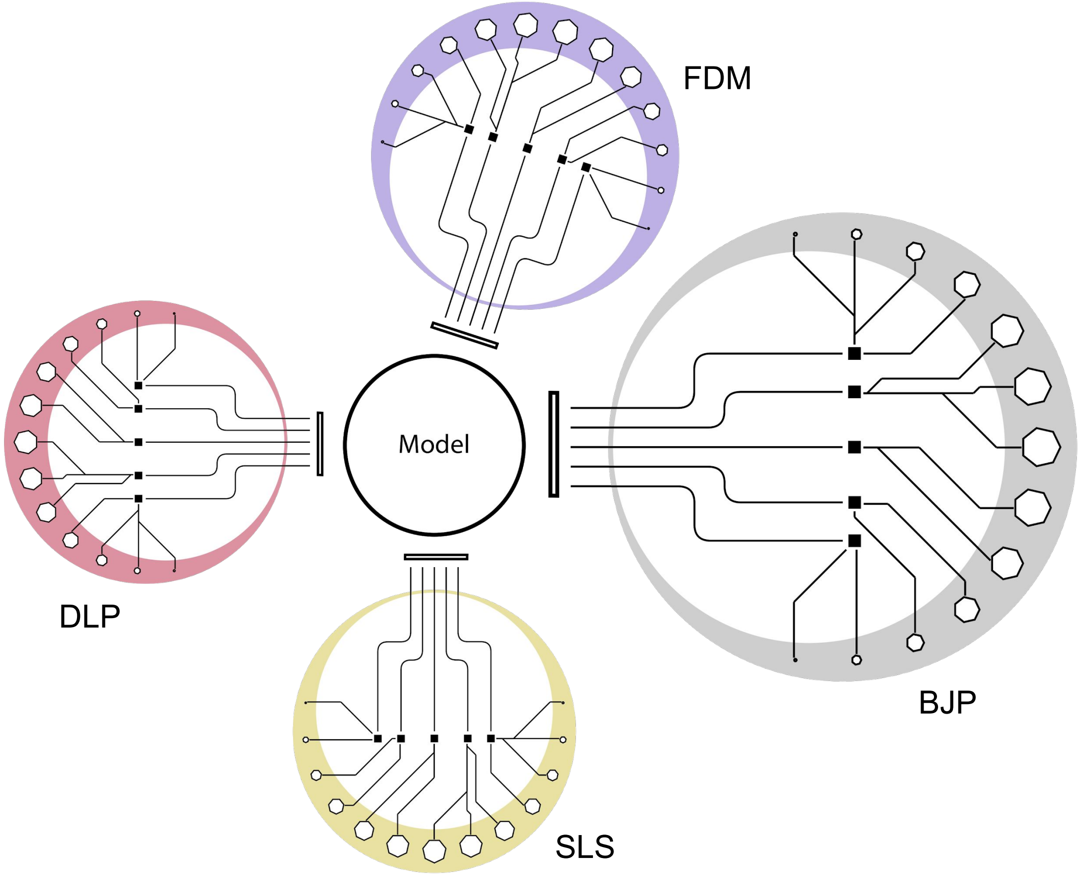
- Retrieve data from model
- Enrich it with process specific information
- Pass it over to hardware

AM Techniques
Classification



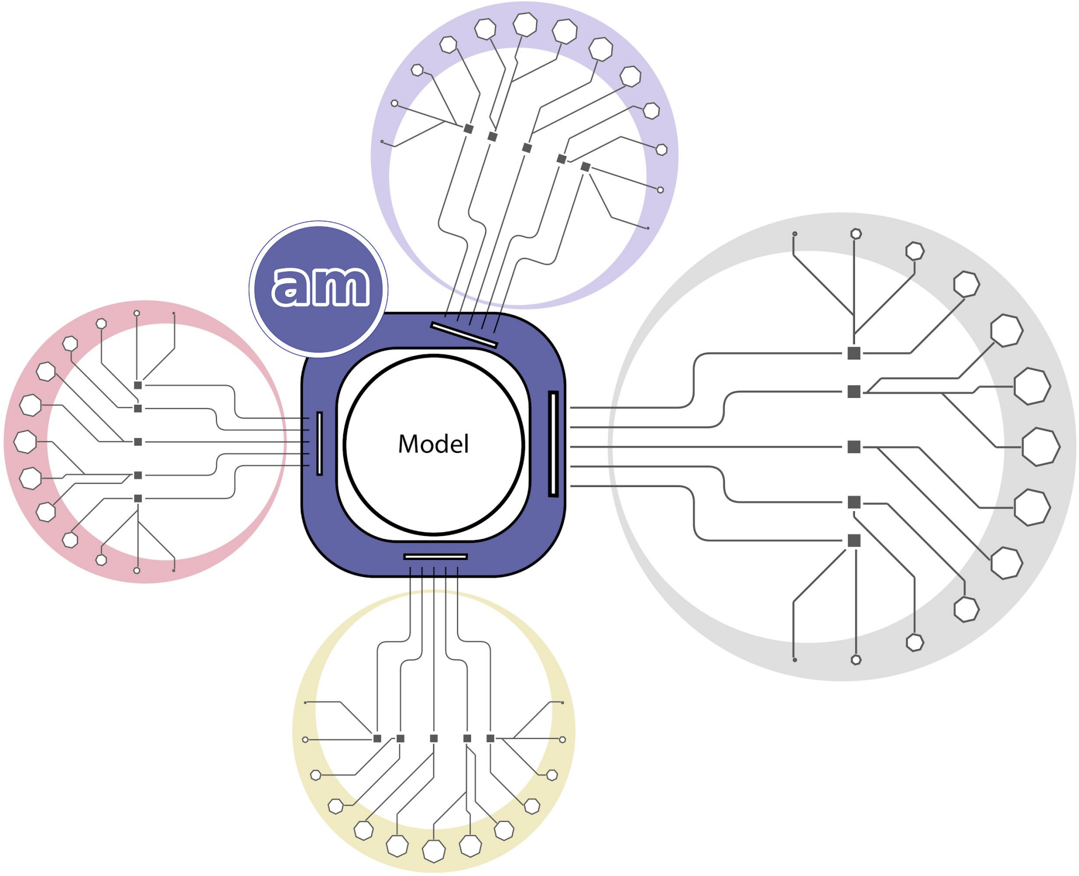
AM Techniques

Common model

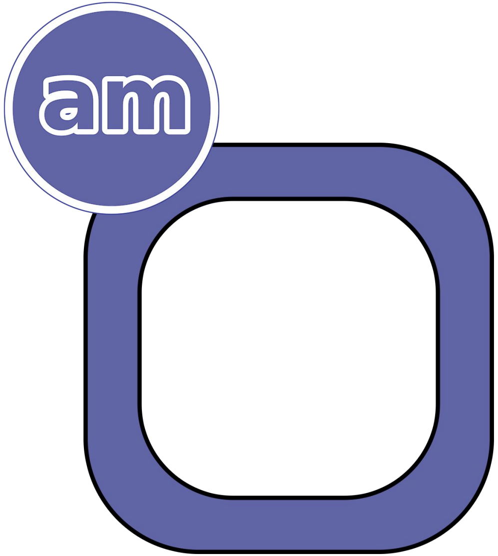


Back to Compas AM

Interfaces



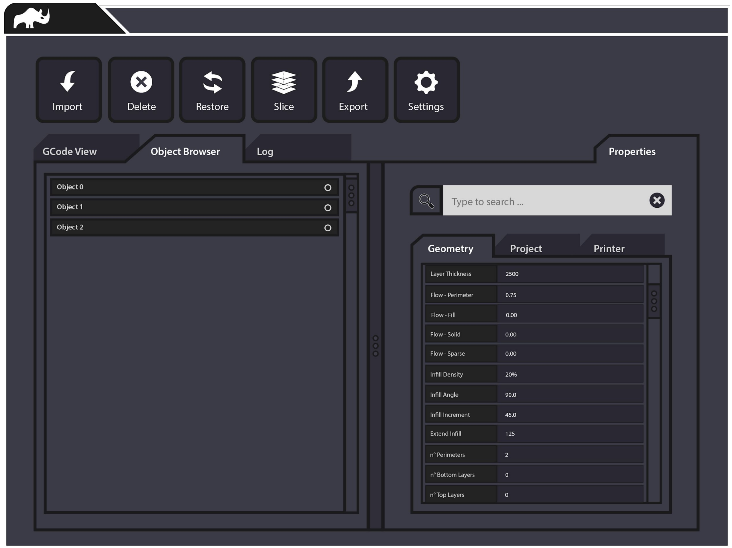
Back to Compas AM
Packages



- Toolpathing tools
- Slicing tools
- Process control
- Fabrication data
- User interaction

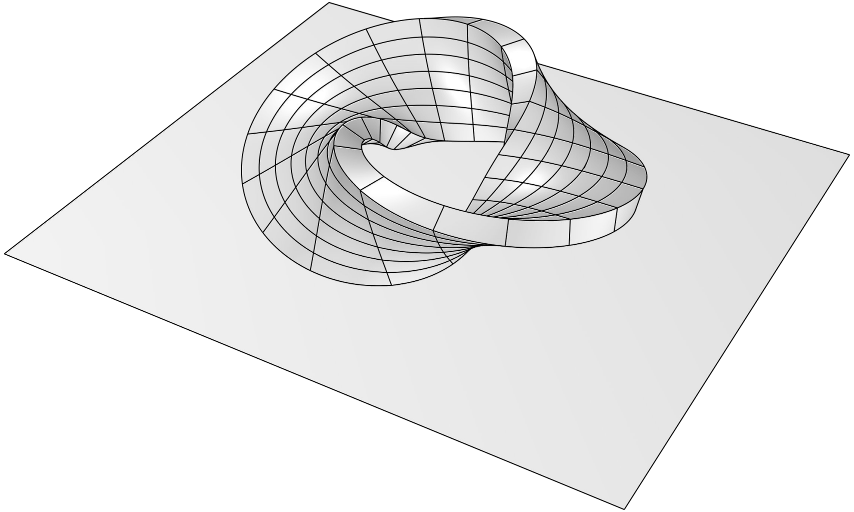
User interface

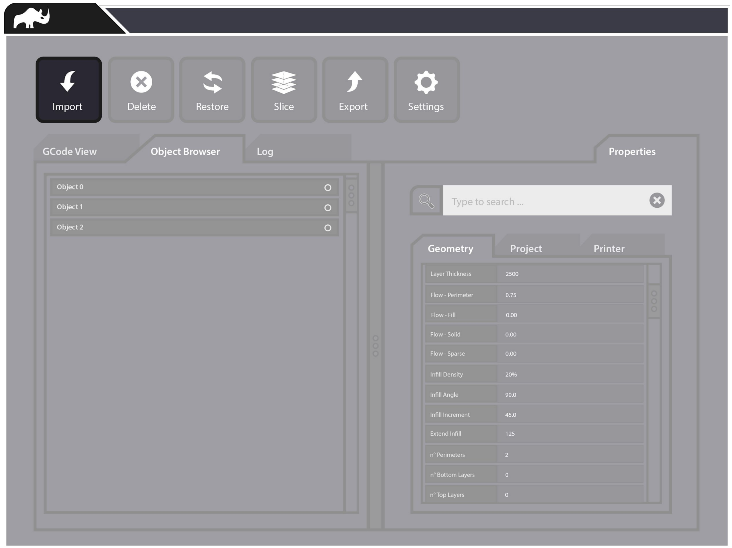
Alpha version



User interface

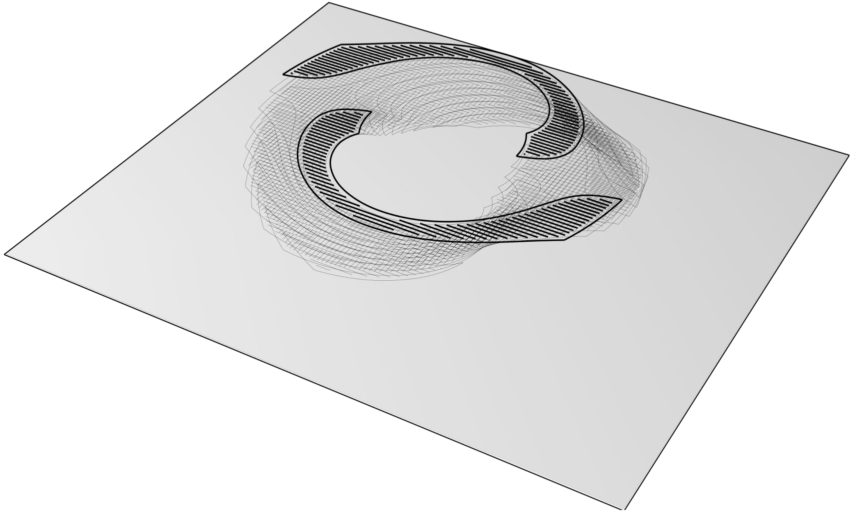
Import geometry

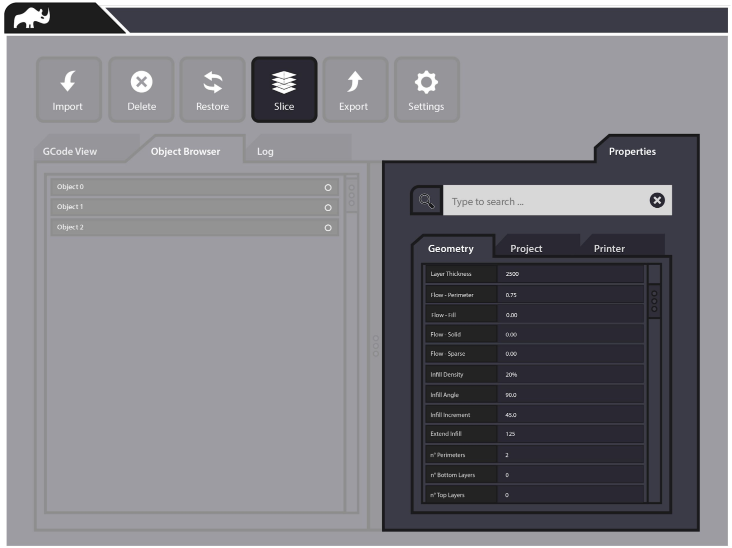




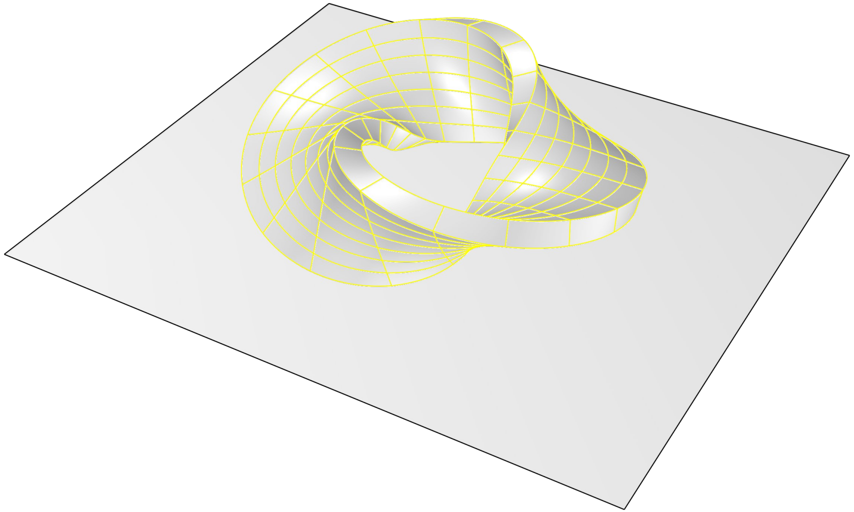
User interface

Process parameters



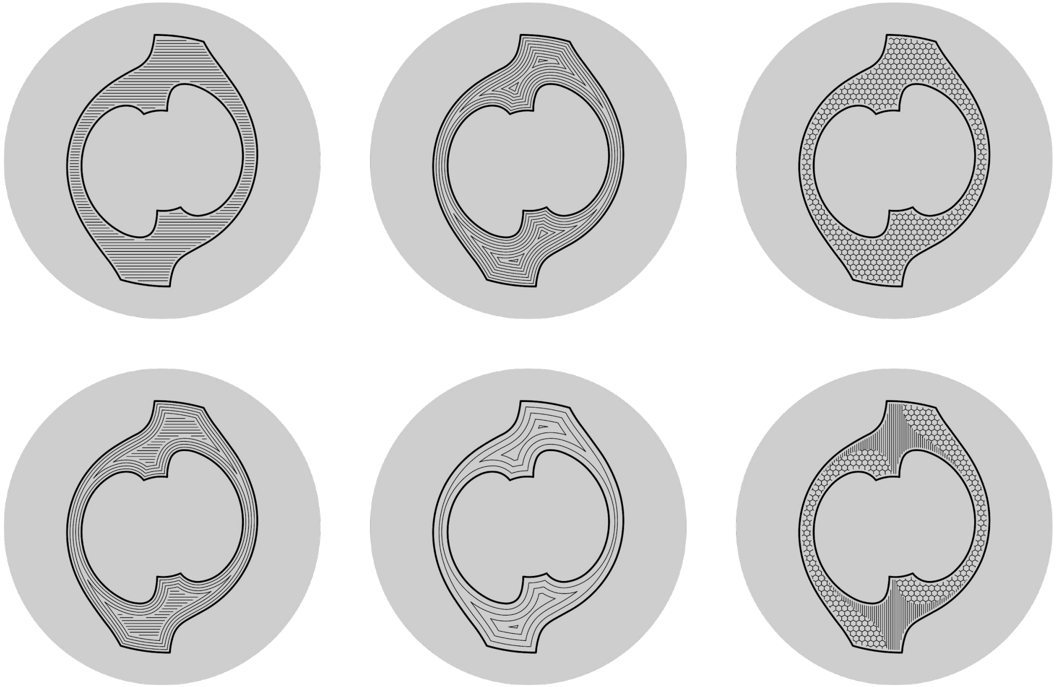


User interface
Object browser



User Interface

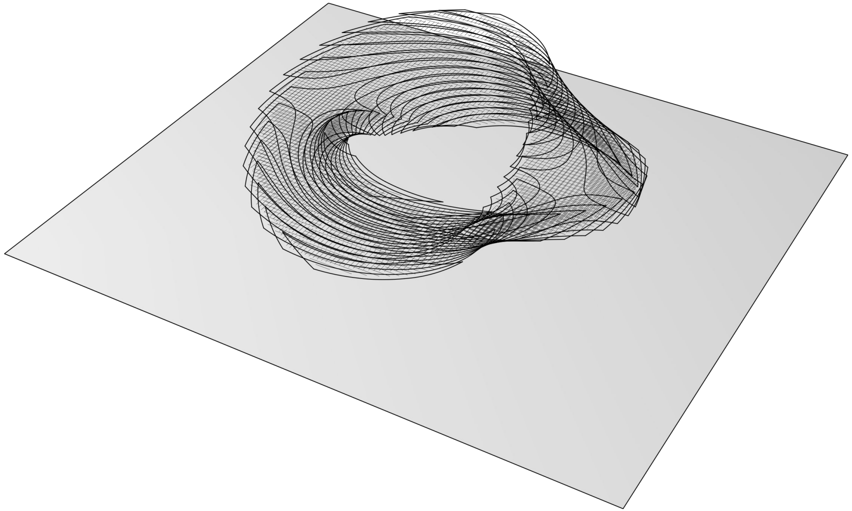
Features

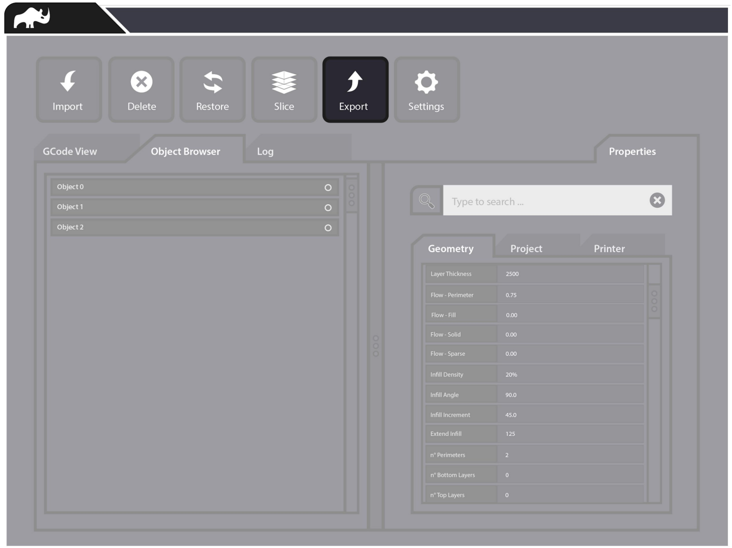


- Toolpath types
- Infill patterns
- Shell thickness
- Density
- etc.

User interface

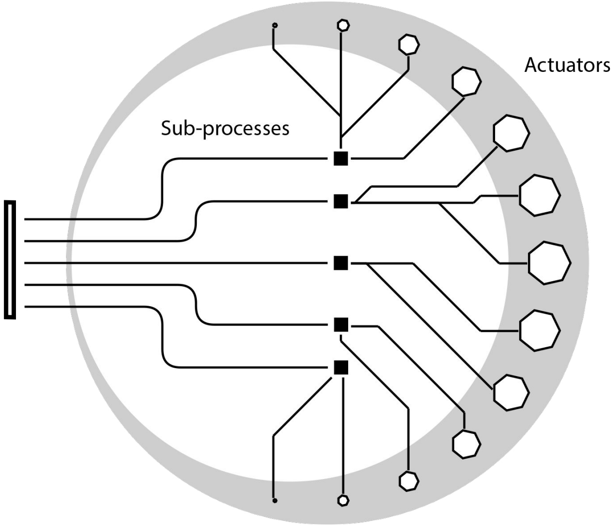
Fabrication data

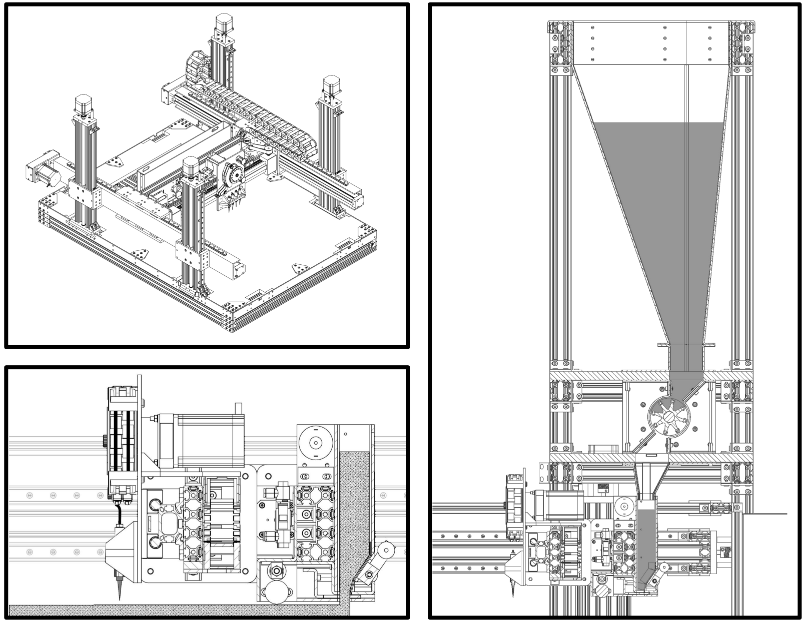




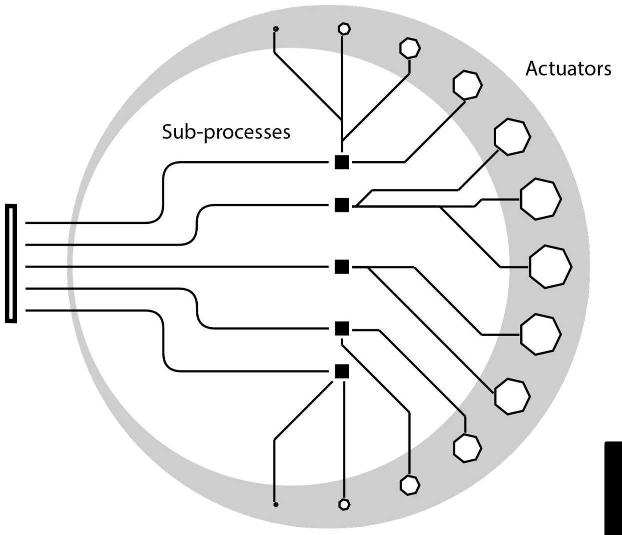
Framework

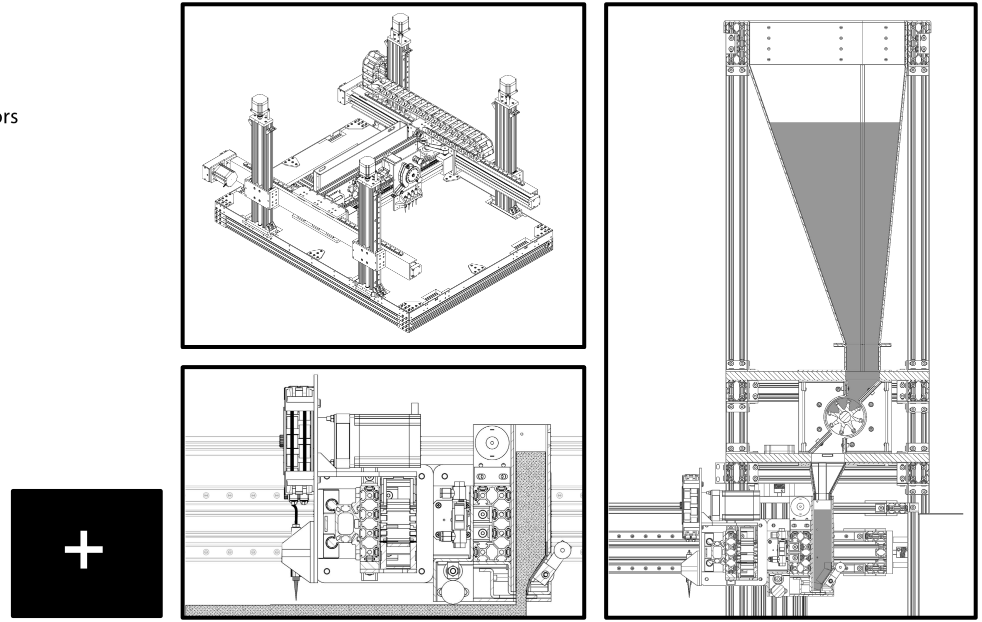
Hardware configuration





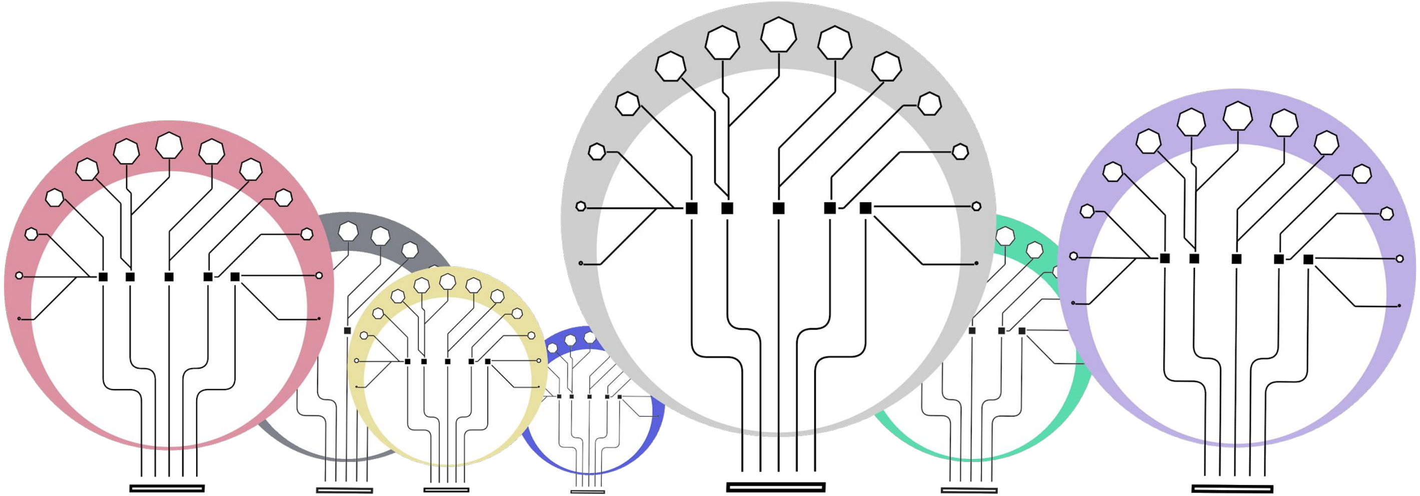
Framework
Hardware configuration





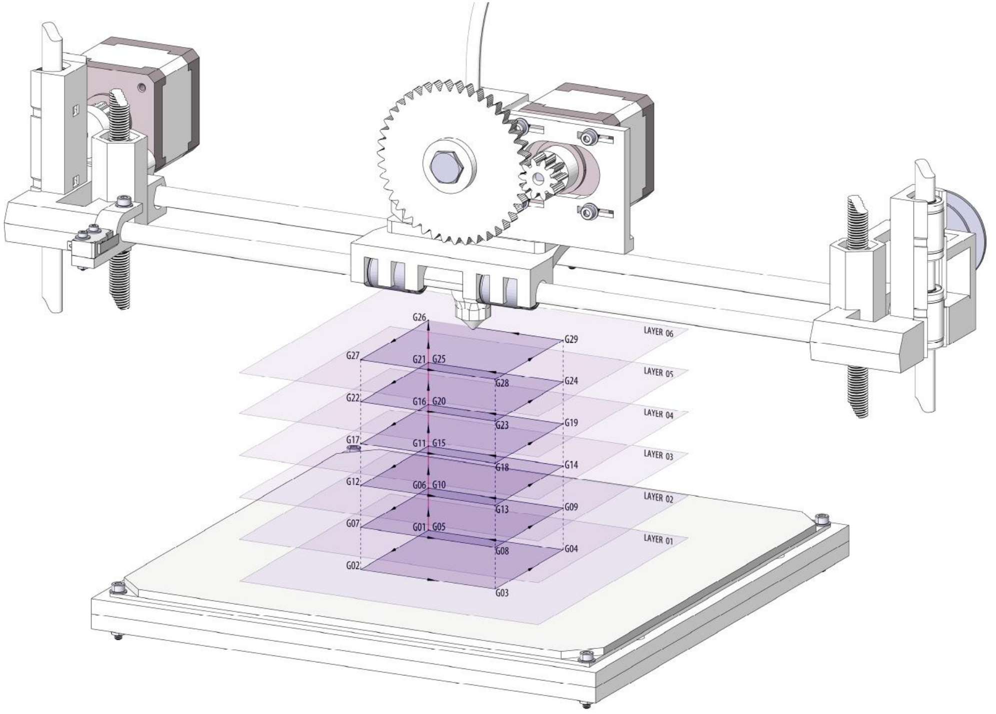
AM Ecosystem

Further development



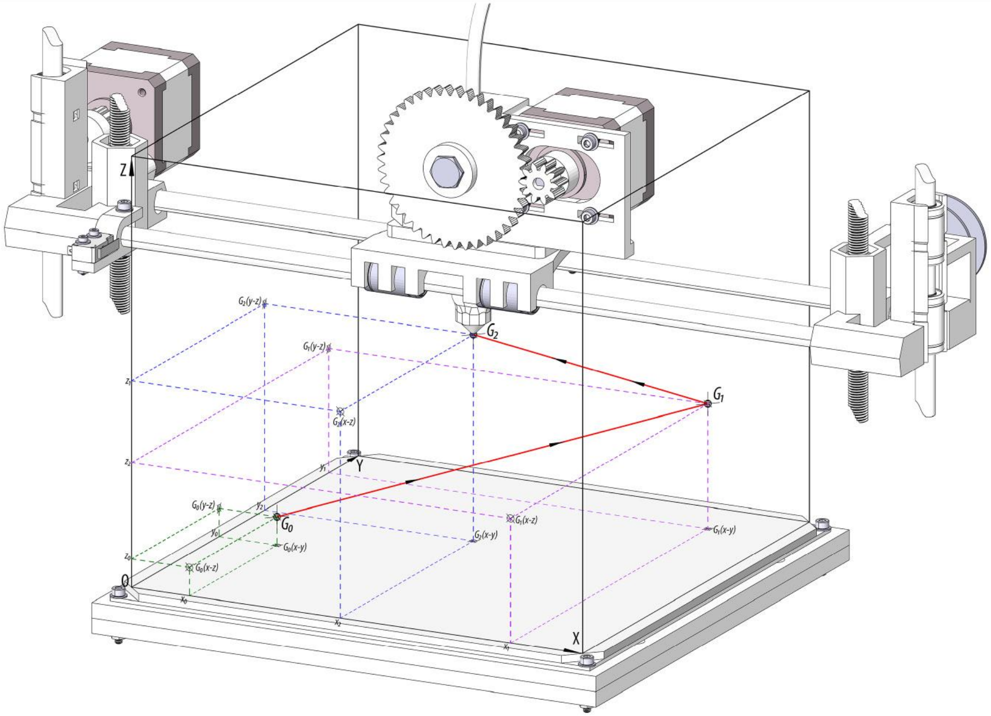
Compas AM

3D Printing module



Compas AM

3D Printing module



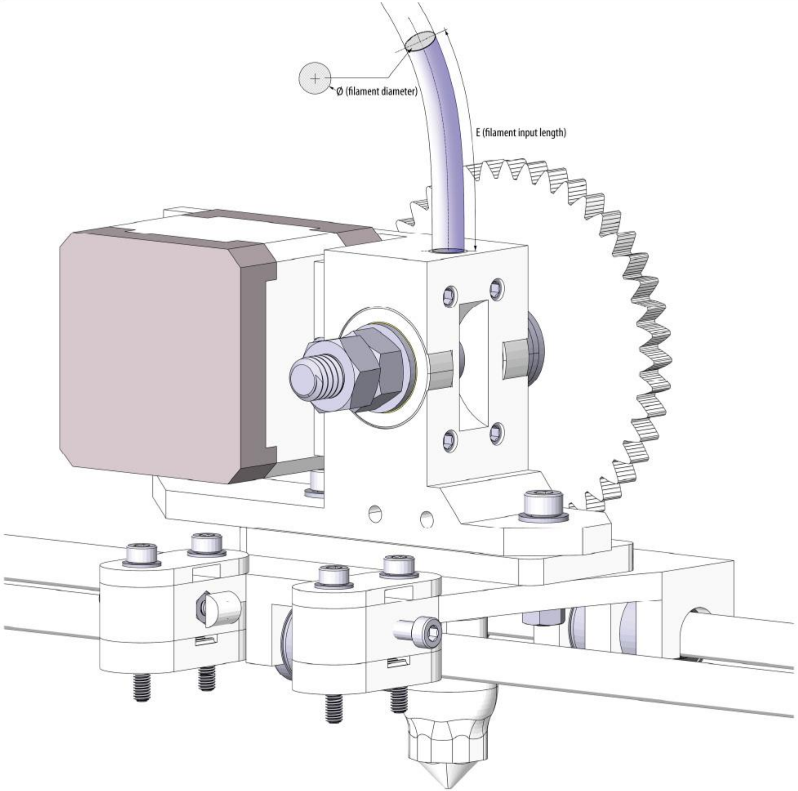
M109 S205 ;set temp.

G1 F3000 ;set feedrate @50mm/s

G1 Xx_1 Yy_1 Zz_1 Ee_1

G1 Xx_2 Yy_2 Zz_2 Ee_2

...



Fv (filament volume) $= \pi \times Ø^2$ *(filament diameter)* $\times$ *Lf (filament length) / 4*

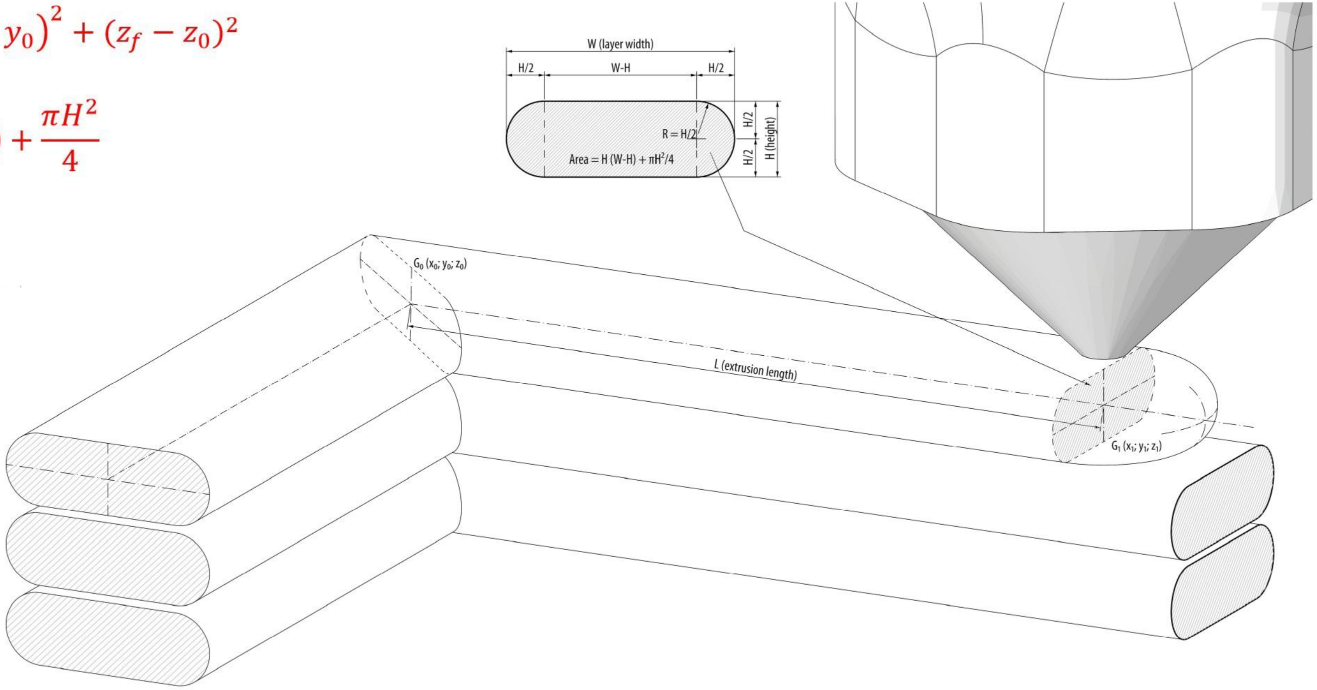
The filament length is precisely what we need to calculate, the E value, therefore, in the previous formula:

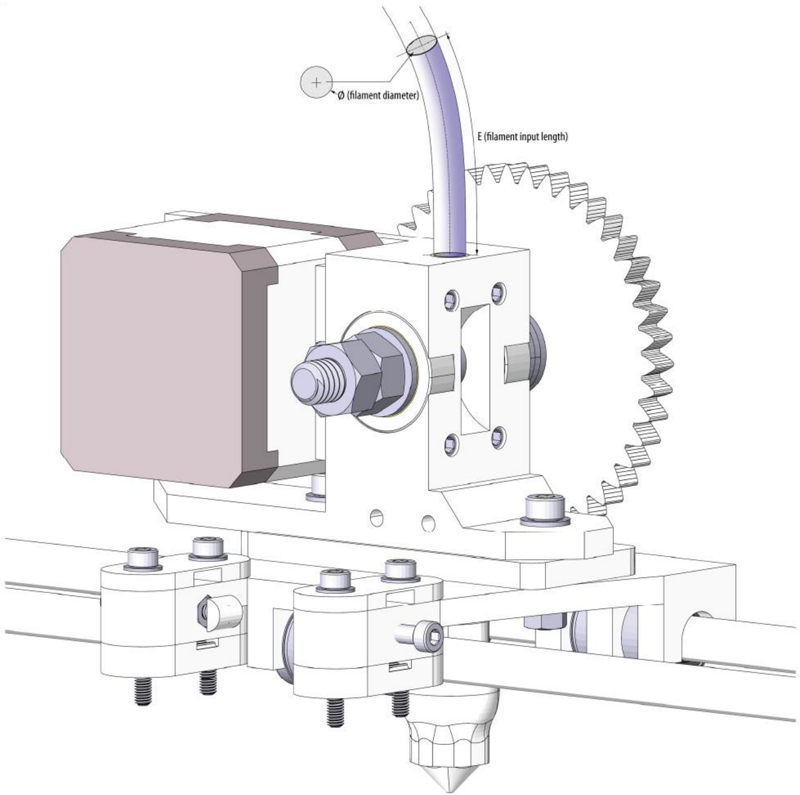
$Vp = Vf$

Vp (printed volume) = A (Layer Section Area) × L (Extrusion Length)

$$L = \sqrt{\left(x_f - x_0\right)^2 + \left(y_f - y_0\right)^2 + (z_f - z_0)^2}$$

$$A = H(W - H) + \frac{\pi H^2}{4}$$





The other requirement for calculating the E value is Vf, the filament volume:

Fv (filament volume) = $\pi \times \text{Ø}^2$ (filament diameter) × Lf (filament length) / 4

The filament length is precisely what we need to calculate, the E value, therefore, in the previous formula:

Vp = Vf

$$L[H(W-H)+\frac{\pi H^2}{4}] = \frac{\pi \text{Ø}^2}{4} \times E$$

$$\boxed{E = [H(W-H)+\frac{\pi H^2}{4}]\frac{4\sqrt{(x_f-x_i)^2+(y_f-y_i)^2+(z_f-z_i)^2}}{\pi \text{Ø}^2}}$$

Where:

Where:	E	- [mm] - E-value (length of filament fed);
	H	- [mm] - layer height;
	W	- [mm] - layer width;
	Ø	- [mm] - filament diameter;
	x_i, y_i, z_i	- [mm] - coordinates of initial toolhead position;
	x_f, y_f, z_f	- [mm] - coordinates of toolhead destination;

A simplified version of the above formula can be obtained by approximating the layer section to a perfect rectangle:

$$Vp = Vf$$

$$L \times H \times W = \frac{\pi \text{Ø}^2}{4} \times E;$$

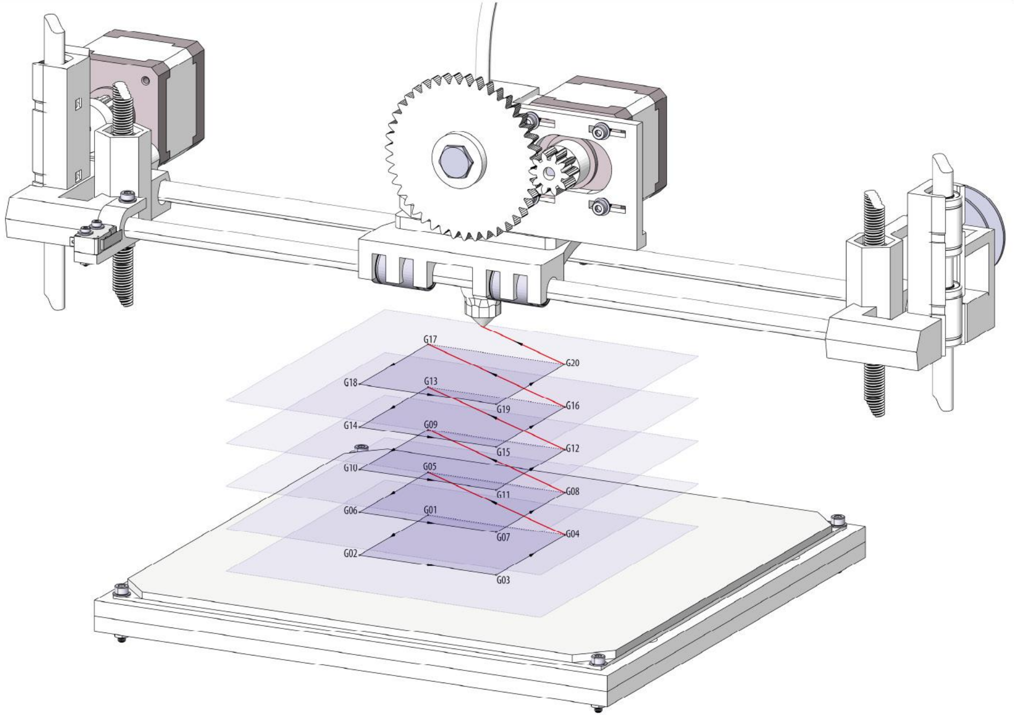
$$E = \frac{4 \times L \times H \times W}{\pi \text{Ø}^2}$$

$$E = \frac{4HW}{\pi \text{Ø}^2}\sqrt{(x_f-x_i)^2+(y_f-y_i)^2+(z_f-z_i)^2}$$

dbt ETH zürich

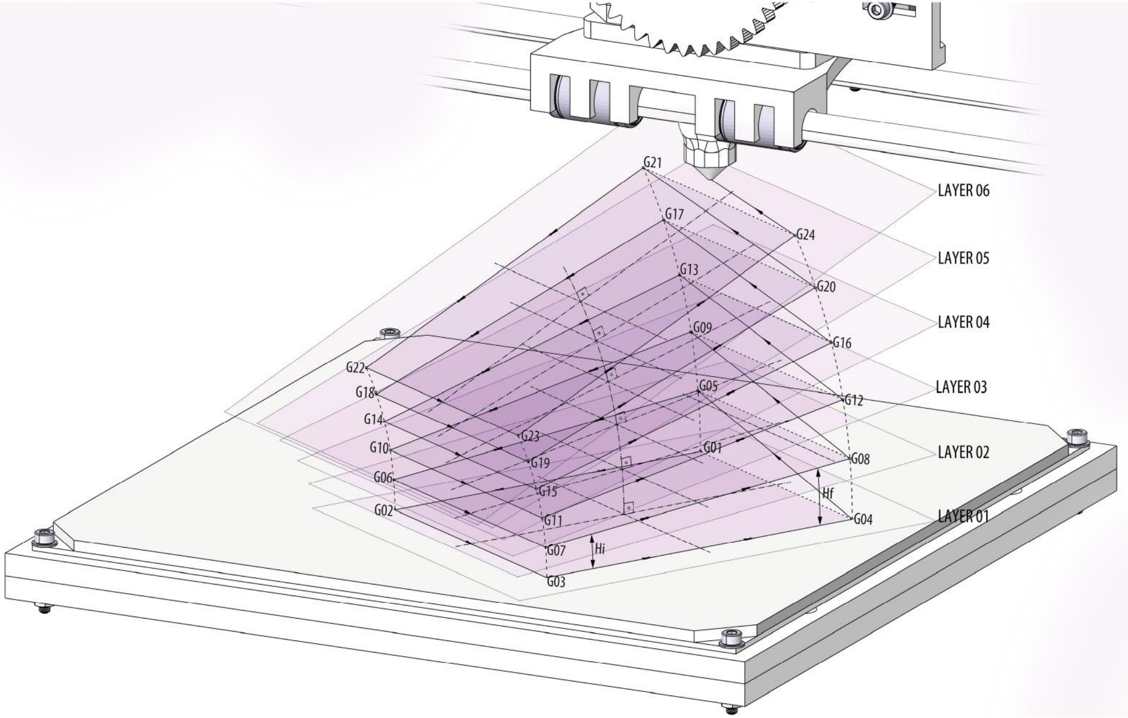
Compas AM

3D Printing module



Compas AM

3D Printing module

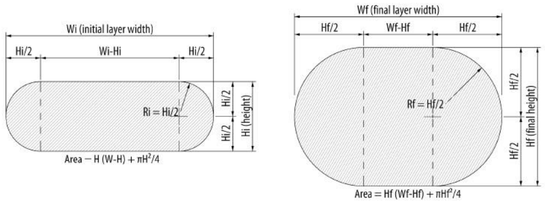


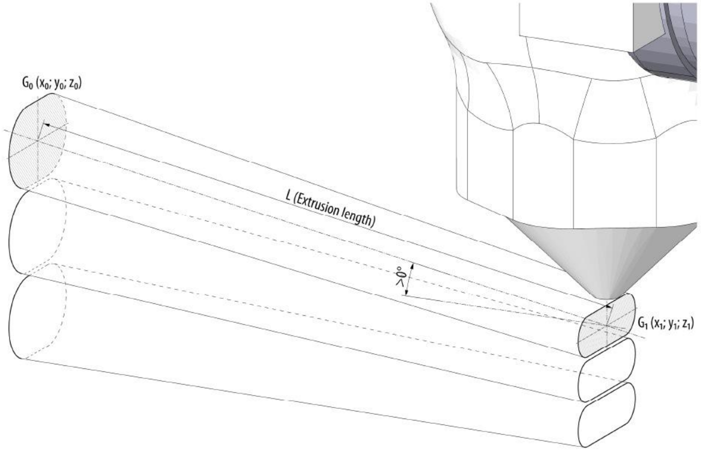
dbt ETH zürich

Compas AM

3D Printing module

```
Private Sub gCode(ByVal x, y, z As List(Of Double), ByVal layW As Object, ByVal ilH As Double, ByVal filD, zTol As Double, ByRef A As Object)
'x, y and z – ordered lists of point coordinates; layW – layer width; ilH – initial layer Height; filD – filament diameter; zTol – tollerance; A – output gcode
Dim gPath As New List (Of String) 'temporary storage for gcode
Dim steps, j, i, nPoly As New Integer
Dim tmPath As String 'temporary storage for gcode lines
Dim tmpX, tmpY, tmpZ, reL,tmpE, tmpEi,tmpEf, iH, fH, layH As New Double
nPoly =  5 'number of instructions in the current layer
'generate the first gCode instruction:
tmPath = "G1 X" & FormatNumber(x(0), 3) & " Y" & FormatNumber(y(0), 3) & " Z" & FormatNumber(z(0), 3)
gPath.Add(tmPath)
For i = 1 To nPoly
      reL = ((x(i) - x(i - 1)) ^ 2 + (y(i) - y(i - 1)) ^ 2 + (z(i) - z(i - 1)) ^ 2) ^ 0.5 'calculate layer length in the current instruction
      layH = ilH 'initialize layer height
      tmpE = 4 * reL * layH * (layW - layH + (Math.PI * layH ) / 4) / (math.PI * (filD ^ 2))'initialize instantaneous E value
      'generate the second instruction:
      tmPath = "G1 X" & FormatNumber(x(i), 3) & " Y" & FormatNumber(y(i), 3) & " Z" & FormatNumber(z(i), 3) & " E" & FormatNumber(tmpE, 3) ' & " D=" & reL
      gPath.Add(tmPath)
    Next
For i = nPoly + 1 To x.Count - 1
      reL = ((x(i) - x(i - 1)) ^ 2 + (y(i) - y(i - 1)) ^ 2 + (z(i) - z(i - 1)) ^ 2) ^ 0.5'calculate layer length for the current instruction
      iH = z(i - 1) - z(i - nPoly -1)'calculate initial layer height
      fH = z(i) - z(i - nPoly)' calculate final layer height
      steps = Int(Math.Abs(fH - iH) / zTol) 'calculate number of discrete steps for the current to achieve desired tollerance
      If steps <= 0 Then
        steps = 1
      End If
      'calculate initial and final instantaneous E value
      tmpEi = 4 * reL * iH * (layW - iH + (Math.PI * iH) / 4) / (math.PI * (filD ^ 2))
      tmpEf = 4 * reL * fH * (layW - fH + (Math.PI * fH) / 4) / (math.PI * (filD ^ 2))
      'calculate instantaneous E values and x, y, z coordinates for each step based on the initial and final values
      For j = 1 To steps
        tmpE = ((steps - j + 1) * tmpEi + (j - 1) * tmpEf) / steps
        tmpX = x(i - 1) + j * (x(i) - x(i - 1)) / steps
        tmpY = y(i - 1) + j * (y(i) - y(i - 1)) / steps
        tmpZ = z(i - 1) + j * (z(i) - z(i - 1)) / steps
        tmPath = "G1 X" & FormatNumber(tmpx, 3) & " Y" & FormatNumber(tmpy, 3) & " Z" & FormatNumber(tmpz, 3) & " E" & FormatNumber(tmpE, 3)
        gPath.Add(tmPath)
      Next
    Next
    A = gPath
```

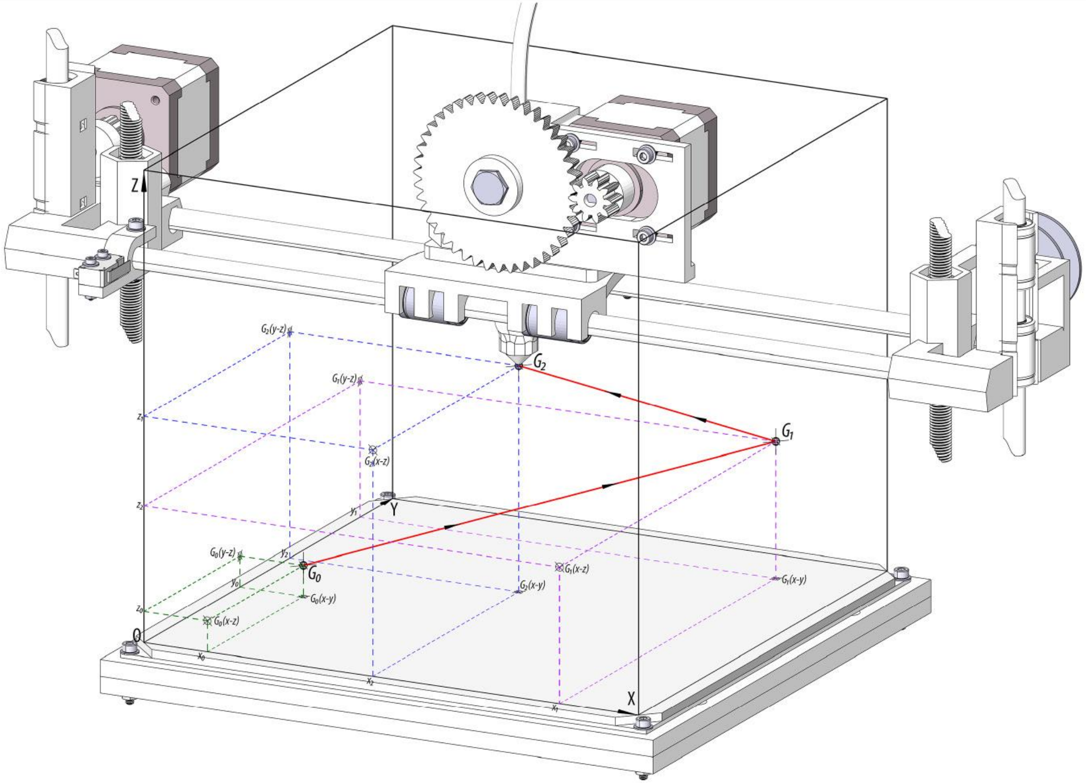




Linear vs. non-linear

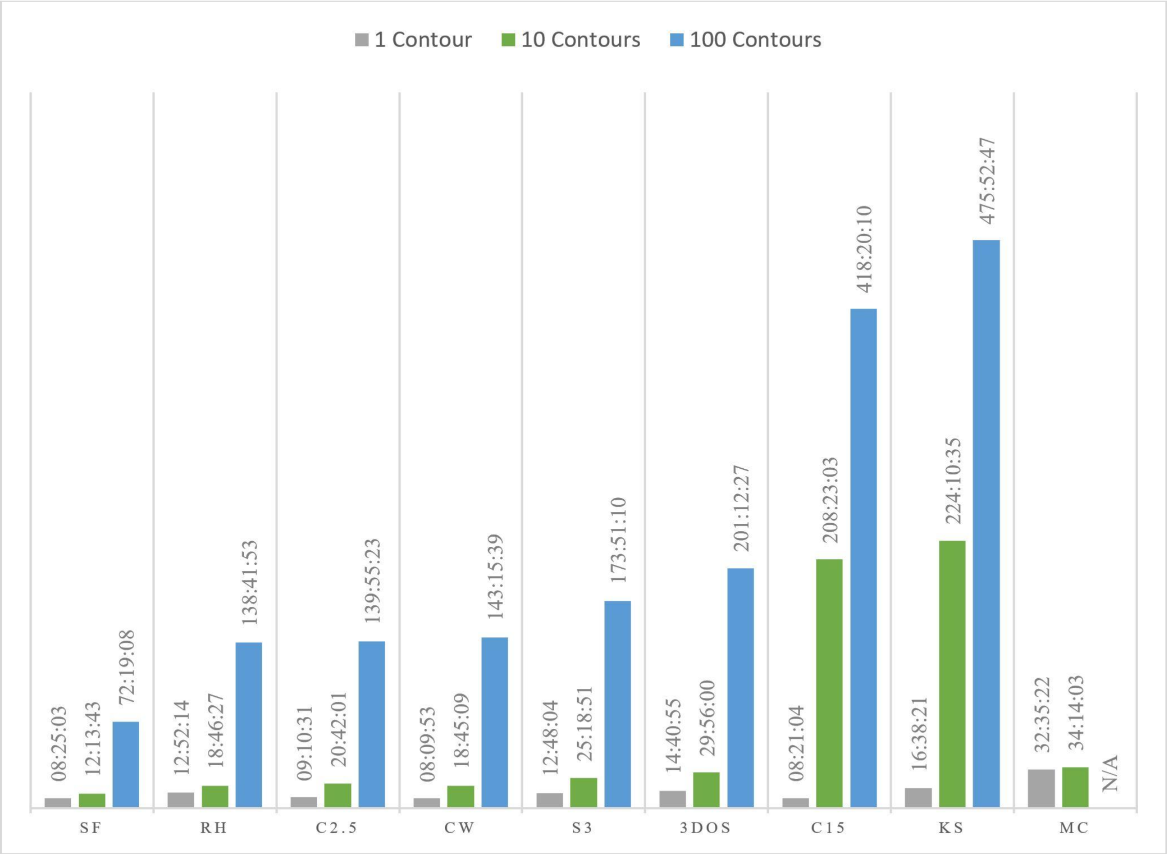
Compas AM

3D Printing module



Each line is a member of a class with attributes:

- Flow rate
- Feed rate
- Retraction / Z/hop

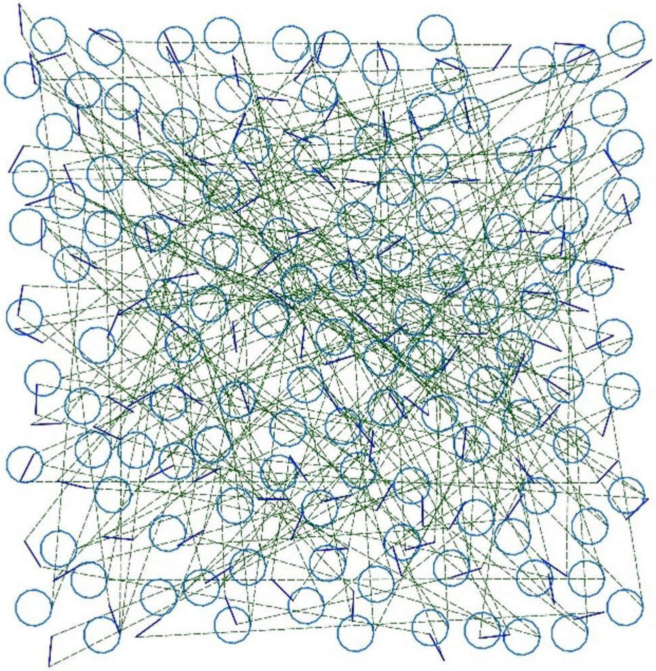


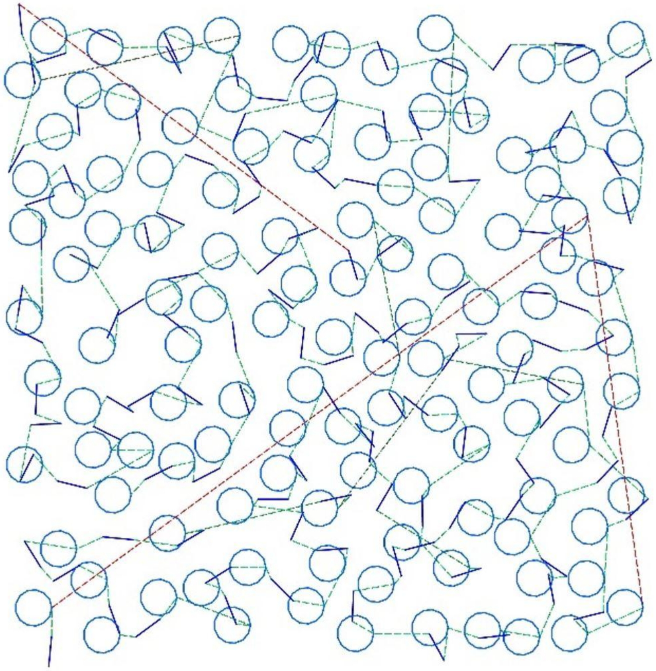
Printing times (in hours:minutes:seconds) for three different digital models with 1, 10 and 100 contours each were used for benchmarking.

Repetier Host 2.0.1 (RH)
Cura 2.5x64 (C2.5)
Craftware 1.14 (CW)
Slic3r 1.2.9 (S3)
3DPrinterOS.com (3DOS)
Cura 15.04 (C15)
KissSlicer 1.5x64 (KS)
MatterControl 1.4 (MC)

Travelling salesman

- NN Algorithm (1974)
- NF Variation (2007)

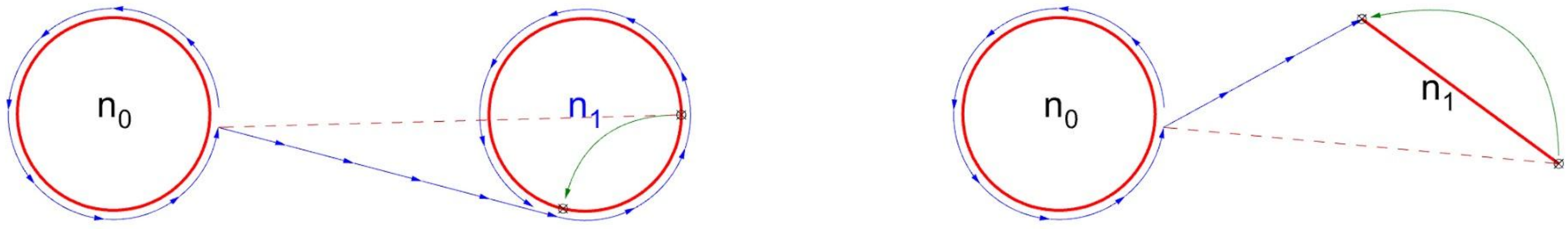




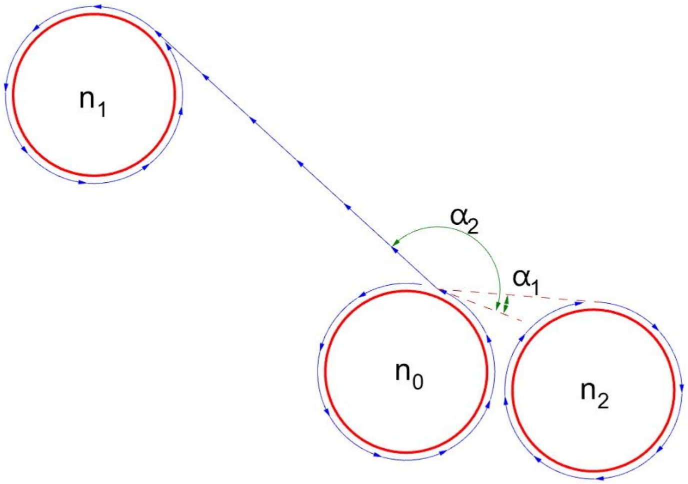
- **Brute force:**

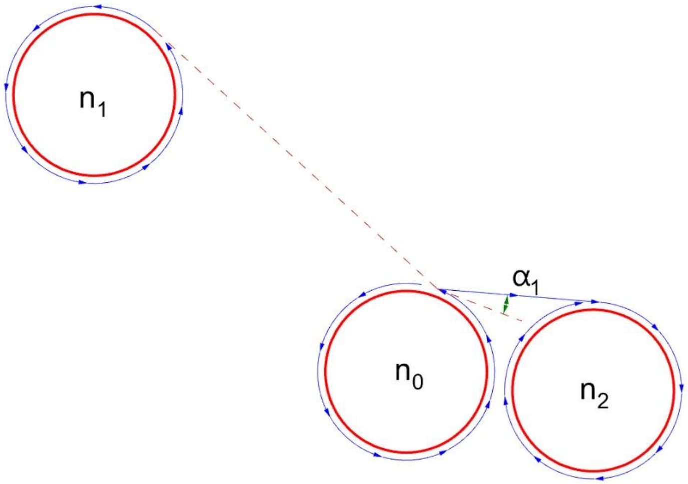
20 contours: 2.432902 e18 permutations

Ray, S. S.; Bandyopadhyay, S.; Pal, S. K. (2007). "Genetic Operators for Combinatorial Optimization in TSP and Microarray Gene Ordering". *Applied Intelligence*. **26** (3): 183–195. CiteSeerX 10.1.1.151.132. doi:10.1007/s10489-006-0018-y.



dbt ETH zürich





α = change in direction in radians
v_{max} = feed-rate of the tool-head in mm/s
a_{max} = maximum acceleration of the tool-head in mm/s^2
j = jerk federate of the tool-head in mm/s.

$d = d_0 + v_0 t + at^2/2$

$v_{bend} = j + \sin(\alpha/2) \times (v_{max} - j)$

$T_{slow} = 2 \times (j + \sin(\alpha/2) \times (v_{max} - j)) / a_{max}$

$D_{slow} = v_{max}\ (2 \times (j + \sin(\alpha/2) \times (v_{max} - j)) / a_{max}) + 2a_{max}((j + \sin(\alpha/2) \times (v_{max} - j)) / a_{max}))^2$

$t_{lost} = t_{slow} - D_{slow} / v_{max}$

$t = 2 \times (j + \sin(\alpha/2) \times (v_{max} - j)) / a_{max} - (2 \times (j + \sin(\alpha/2) \times (v_{max} - j)) / a_{max}) - 2a_{max}((j + \sin(\alpha/2) \times (v_{max} - j)) / a_{max}))^2 / v_{max}$

To do...

- Supports
- Multi-material
- Discretisation
- Orientation
- Nesting
- Interactive design tool
- light version for live feedback

Compas AM
Workflow