Curves -> Points -> X,Y,Z

A NURBS curve could be directly created in Grasshopper® or could be made in Rhinoceros® and then referenced to Grasshopper® with the 'Curve' parameter.

Notice that Rhinoceros® names 'curve' to any linear element making no difference between the type of curve depending on the degree:

degree 1 = line or polyline

degree 2 = circle, ellipse, parabola, hyperbola, arcs.

degree 3 or more = freeform curves

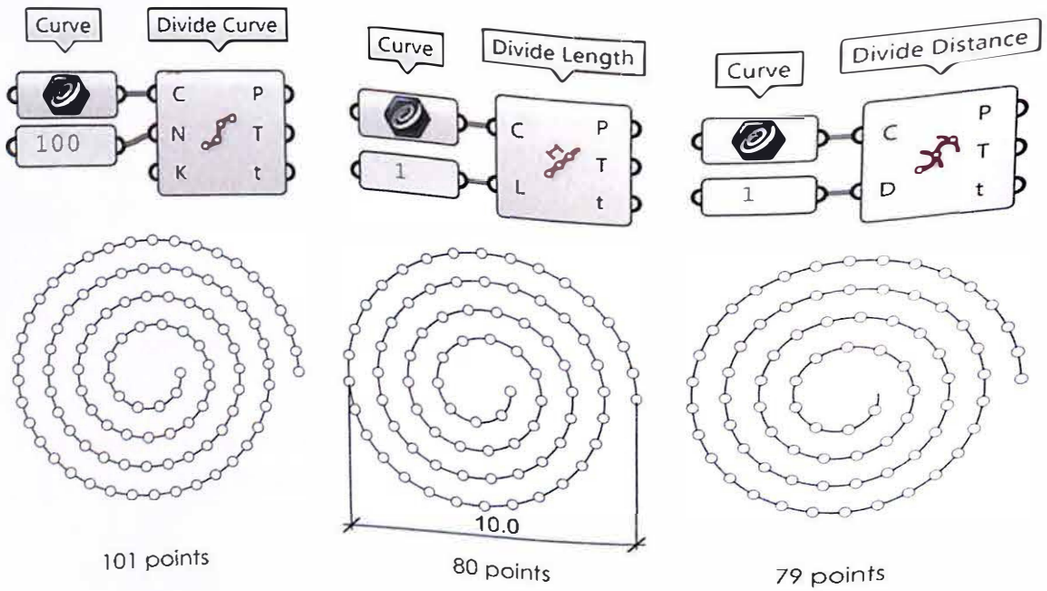
A NURBS curve has infinite points. To translate the geometry of a curve into g-code, we must discretize it into a certain amount of points. The idea is to create the minimum quantity of points possible in order to have a smaller g-code without losing precision and accuracy.

We can follow two strategies:

1. To transform directly the curve into points with a **'Divide'** tool.
2. To transform the curve into a polyline with **'Curvetopolyline'** component and go on as in the previous chapter, *Polylines → Points → X,Y,Z.*

1. 'Divide' tools:

There are different division tools such as 'Dividecurve', 'Dividedistance' or 'Dividelength' to find points on the curve with different results. This strategy would lead us back to the chapter *Points → X,Y,Z.*



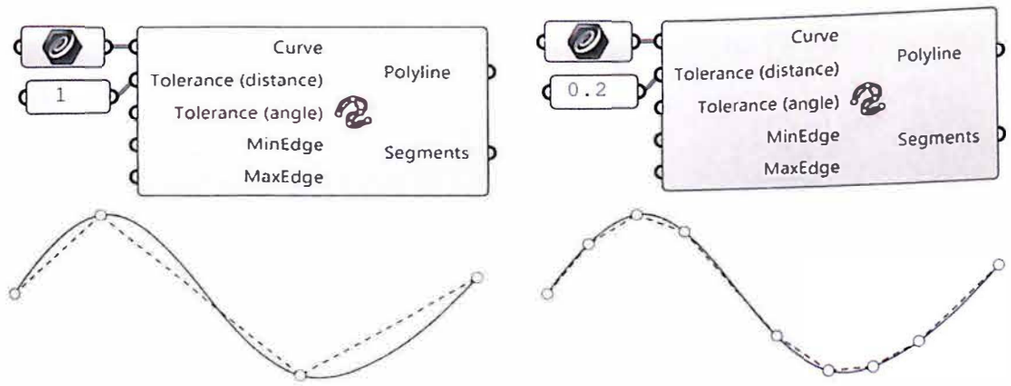
A greater number of points will increase the accuracy from the curve to the points. The Grasshopper® user will probably have some experience with these components.

2. 'Curvetopolyline'

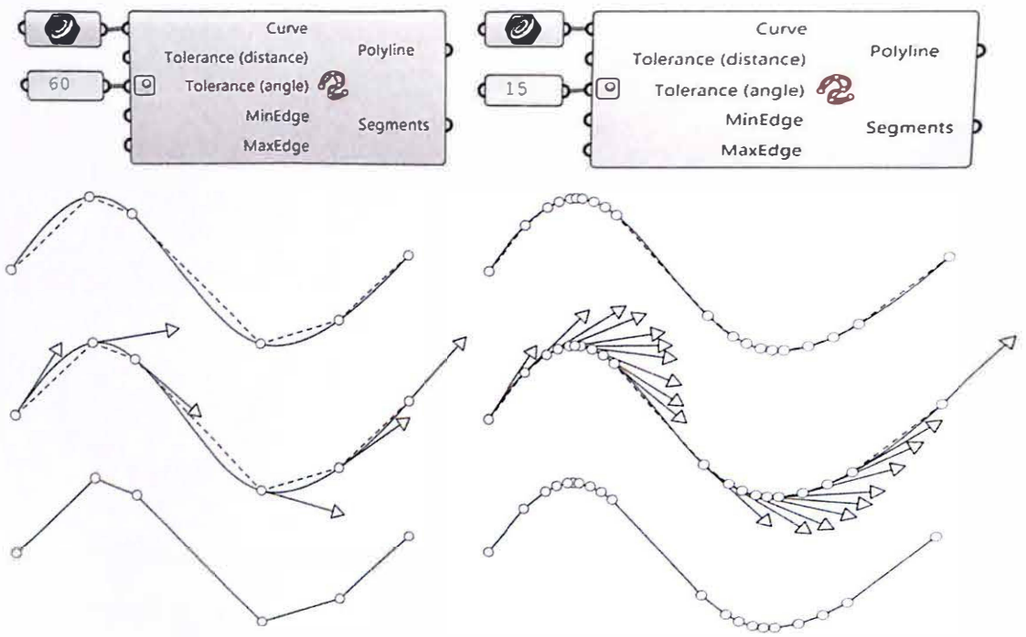
'Curvetopolyline' is a component that transforms a curve into a polyline allowing us greater control of the result, rather than just a simplified division of the curve as with the previous tools. With this tool we will get a polyline and so, we should go on working as in the previous chapter *Polylines → Points → X,Y,Z.*

This component controls the resultant polyline through 4 inputs:

Tolerance (distance): deviation tolerance. This is the maximum distance between the curve and the resultant polyline. The units are the units of the model in Rhino. This means that if we are working in mm, '1' represents 1 mm tolerance. The smaller the distance is, the more accurate the polyline will be.



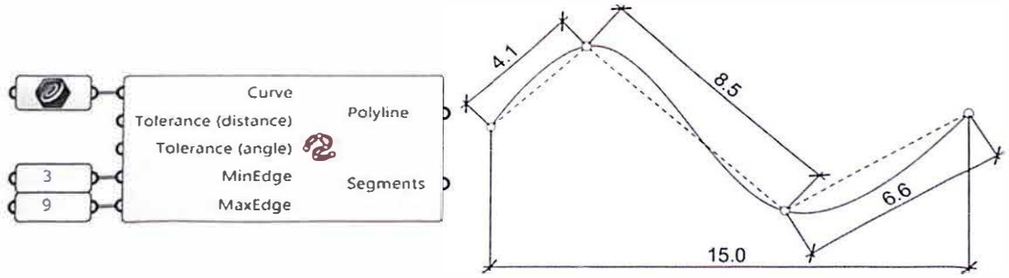
Tolerance (angle): it is the angle tolerance in radians. If the user is not familiar with radians, it is also possible to work in degrees by changing the option with a right click over tolerance. The angle for the tolerance is the maximum angle between the tangent vector to the curve at a point and the polyline at the same point. A smaller angle will create a more accurate polyline.



MinEdge: optional minimum allowed segment length.

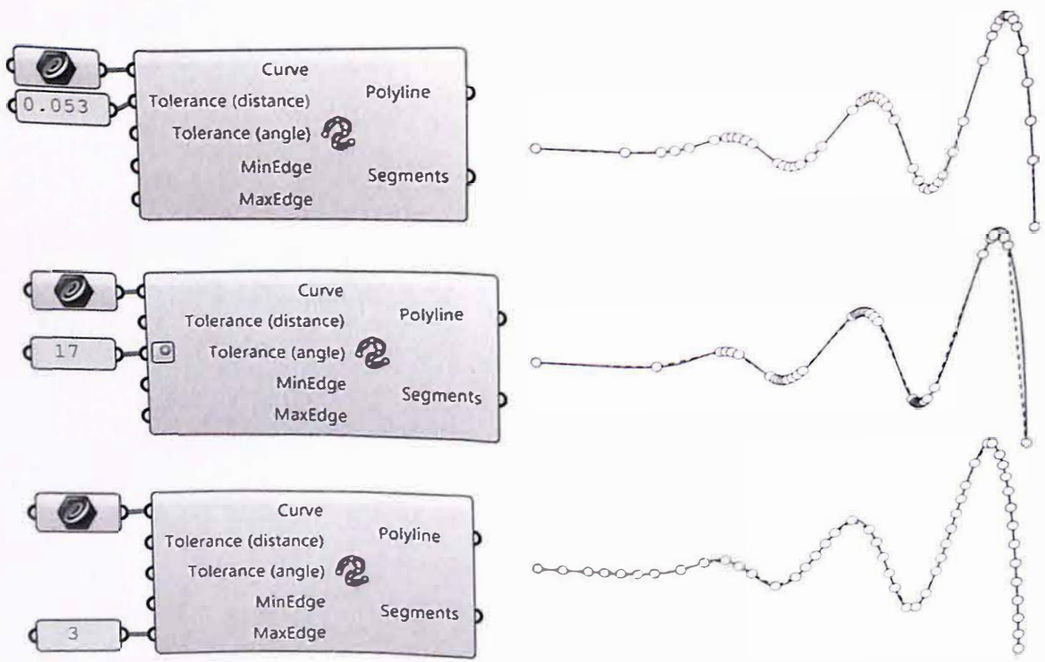
MaxEdge: optional maximum allowed segment length.

Minimum and Maximum edge parameters could be combined to get a polyline within these restrictions. E.g. a polyline with segments of more than 3 and less than 9 mm length:

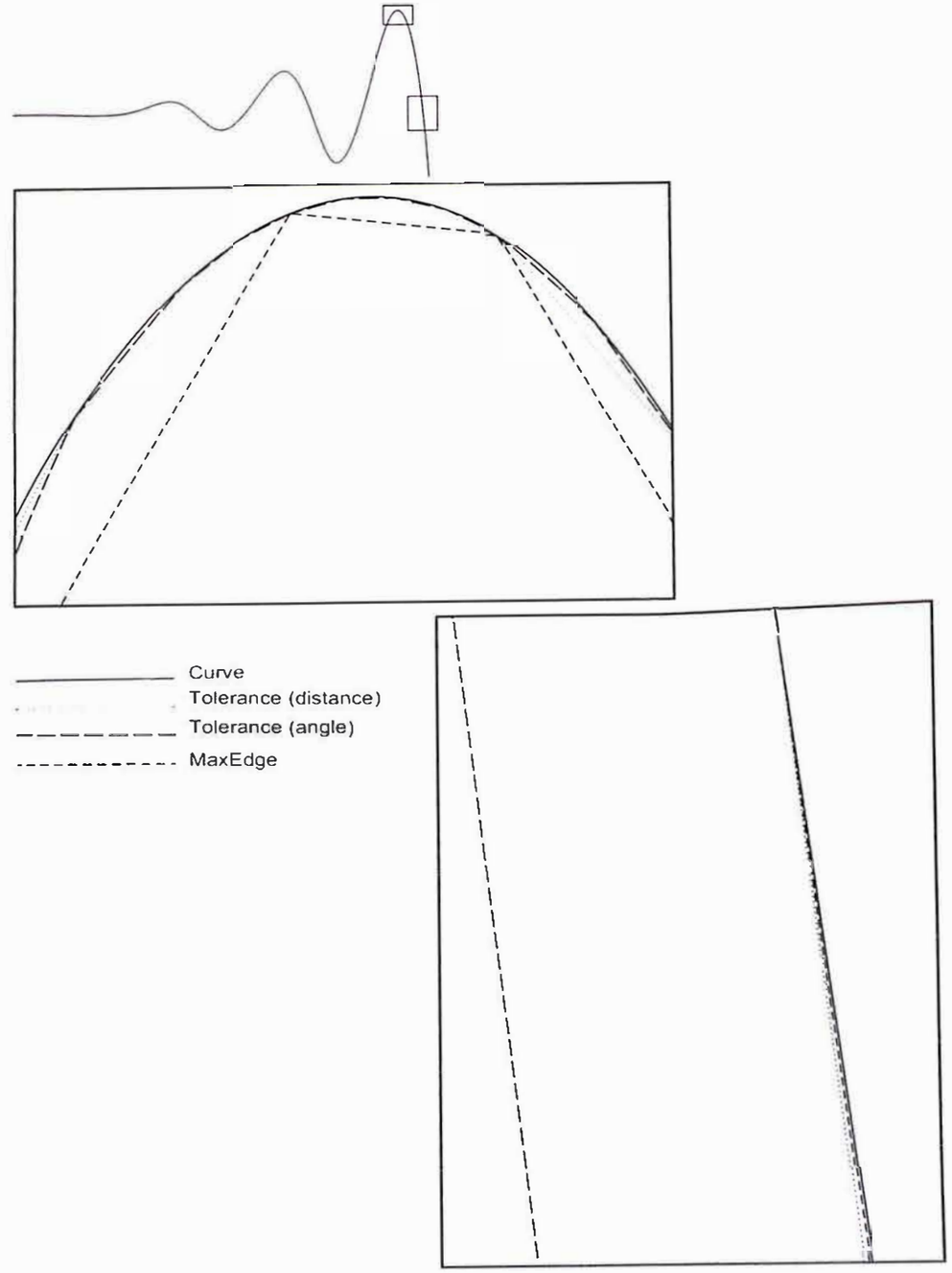


Or, we could just work with the 'MaxEdge' input to get a fitted polyline, which would be equivalent to the 'Dividedistance' component. It looks like we are creating a very dense and fitted polyline but if the curve has varying curvatures, in the low curvature area we will get extra unnecessary points that will lead to a larger g-code file. On the other hand, in the parts of the curve with a big curvature we will not have a good accuracy. In that case it may be better to control the 'Tolerance (distance)' or the 'Tolerance (angle)' inputs.

Mind the difference: with the 'Tolerance' inputs, more points are obtained in the areas of the curve with more curvature, for this example of a curve of 50mm. length in X direction.



The input parameters for these examples are created so that the total amount of vertexes for the polyline is 50. The difference is on the points' disposal. Notice in the next figure how the resultant polyline fits the original curve better depending on its curvature. The 'Tolerance (angle)' works better than the others in the areas with more curvature but does not work on the points in the straight ones. The opposite behaviour happens with the 'Maxedge' option. For the same amount of points, 'Tolerance (distance)' is the most equilibrated option for the different parts of the curve.

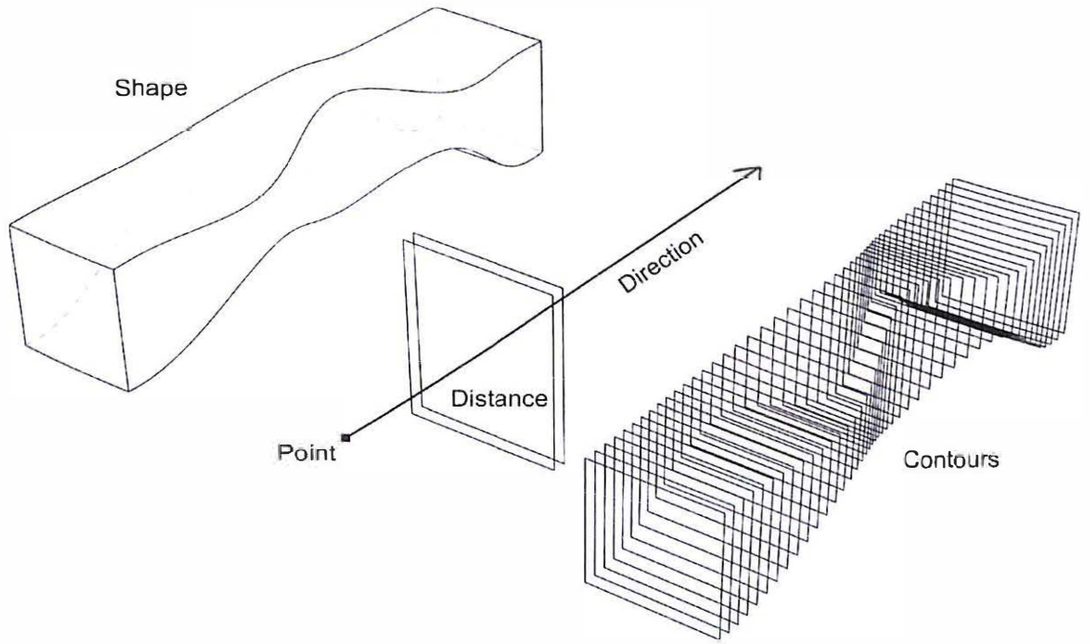


Mesh -> Polylines -> Points -> X,Y,Z

A mesh is a three-dimensional object compounded of vertexes in the cartesian space, joined with edges creating faces that could be triangular, quadrangular or polygonal (n-gons).

To 3d print a mesh following a path, we need to extract curves from it.

A very popular tool is 'Contour'. It makes sections of the 3D object with virtual planes perpendicular to a chosen direction:

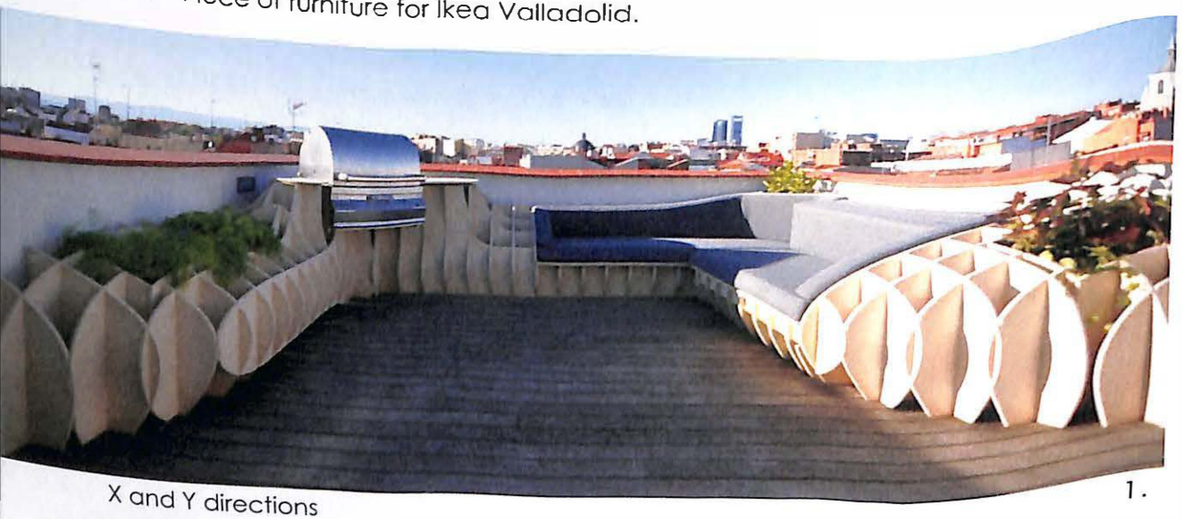


It is widely used in architecture, interior design and furniture design as it allows us to transform a 3dimensional object in 2dimensional planar curves that can be manufactured with standard planar materials used in construction, as wood boards, glass, etc.

Here some examples by Controlmad Advanced Design Center (Madrid).

(Pictures by Controlmad – all rights reserved)

1. Rooftop in Madrid.
2. Rooftop in Madrid.
3. Entrance sculpture for Arup Engineering, Madrid.
4. Piece of furniture for Ikea Valladolid.



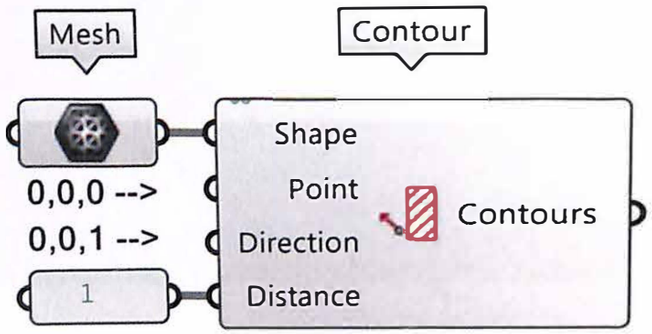
X and Y directions 1.

X and Y directions 2.

3. Z direction

4. X direction

The typical vector direction for architecture or design could be along any given direction; furthermore, contours in two opposite directions are usually combined in order to create a stable and steady structure. In 3d printing the most common direction for the contours is the Z axis.



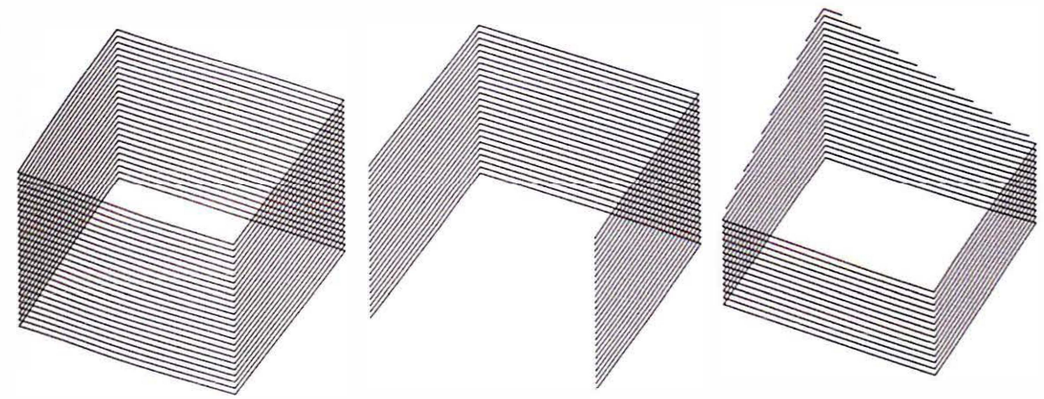
'Shape' input will be the mesh or Brep (surface or polysurface).

'Point' is the start point, by default 0,0,0.

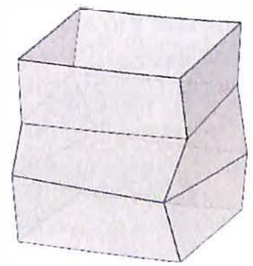
'Direction' is a vector. It represents the normal (perpendicular) direction to the cutting planes and as such, to the resultant section curves. By default, it is '0,0,1' that represents a unitary vector in the Z axis.

'Distance' is the distance in Rhino units (mm.) between sections. For 3D printing it will represent a very important concept as it is the **Layer Height.**

Depending on the geometry of the mesh or Brep, it could generate closed curves, open curves or both:



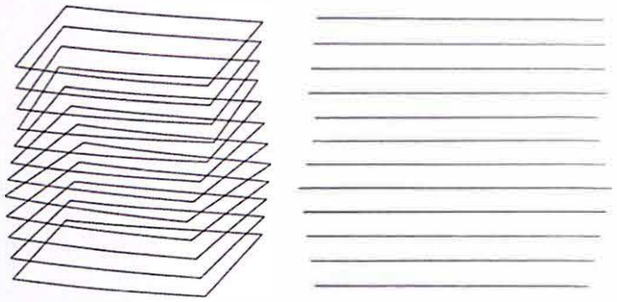
Clay is very sensitive to any change on speed or movement. In order to improve the results on the pieces, it is very important to reduce the amount of travels of the extruder and the changes in extrusion direction to a minimum. This help us to control the seam. The seam is both the origin and the end point of a closed curve. It will determine the order of the points in the list to be followed by the 3d printer. A good control over the seams of the contours will be necessary to improve the results of the printed object. Let's begin with a mesh that produces only closed polylines.



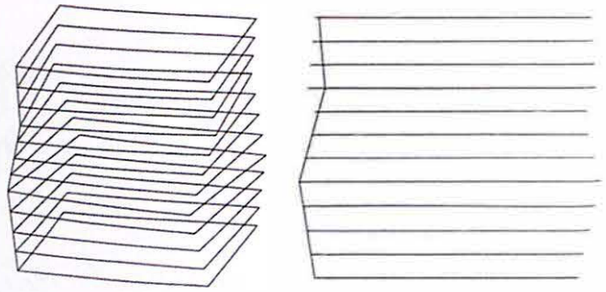
'Contour' component will section the mesh and will output closed polylines. According to the process explained in previous chapters, the printer will follow those sections and will move from one to the next without extruding clay. In this case the stop&go of the extruder could display the seam more than is desired (1).

There is the possibility to extract the points, flatten all in one list and create a continuous polyline. This way we can avoid the extruder stopping but we will have an extra amount of clay in the vertical part of the path (2).

(1)

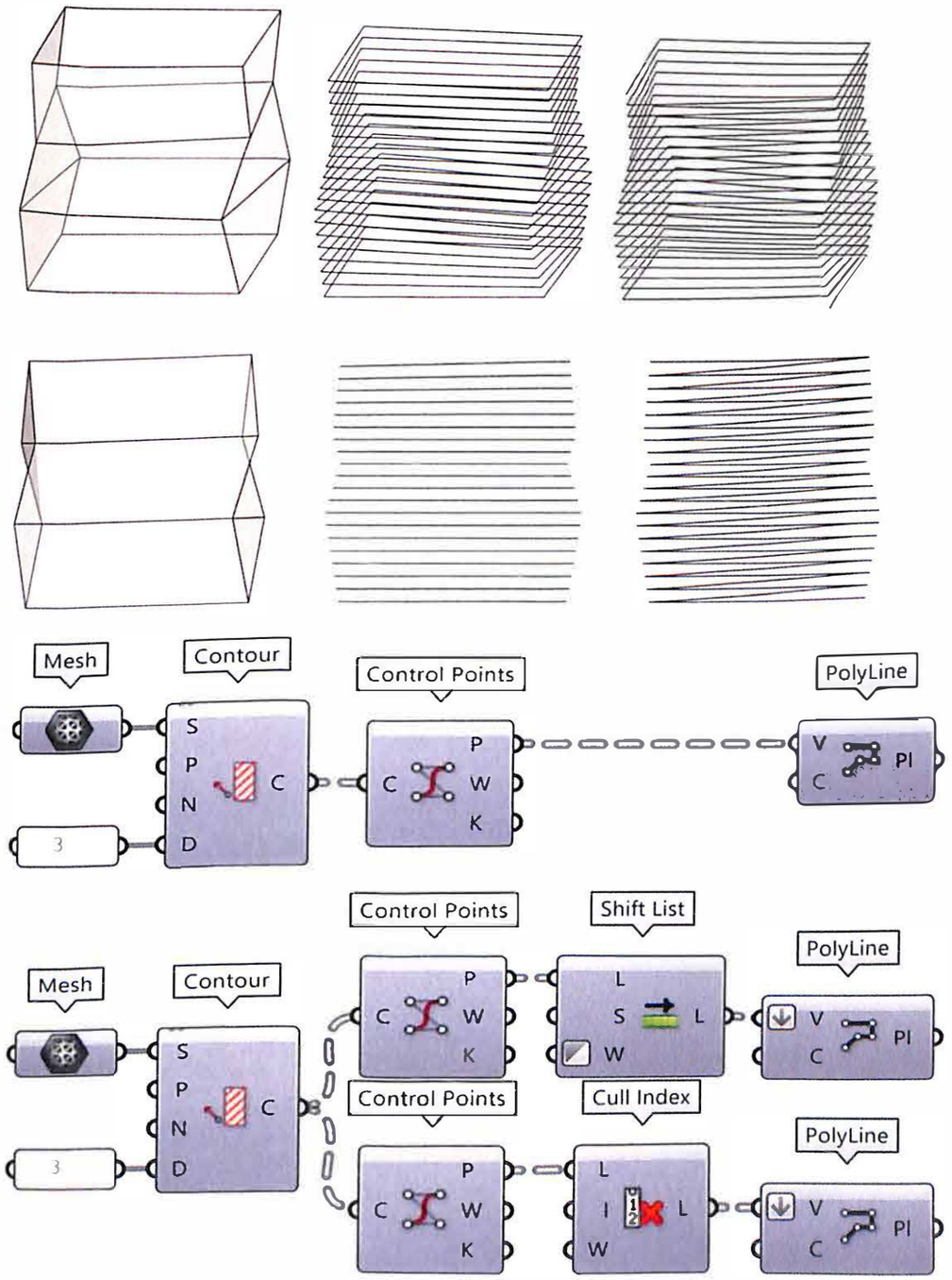


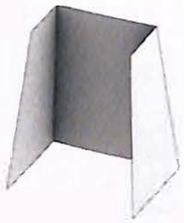
(2)



Potentially, the best option is to create a continuous path connecting the last point of the polyline to the second point of the next by creating a certain slope and so, one continuous polyline (3). To remove the first point in the list, there are two easy options. Use the component 'Shift list' on the points of each contour and then selecting 'Invert' in 'Wrap' input to avoid its addition to the list at its end - or use the tool 'Cull index' entering 0 as integer in the 'indices' input to cull the first element in the list. Then 'Flatten' the list of points to get one single list that creates a single continuous polyline*. This system will be improved in the next chapter with curves, where we can select how many points we want to create along the slope.

* 'Contour' tool creates a data tree with the sections, one list per curve. This makes it mandatory to 'Flatten' the lists of points at the input of 'Polyline' as they come in lists by contours. Flattening the lists, the polyline will go from the last point of one list to the first of the next list as in the following figure:

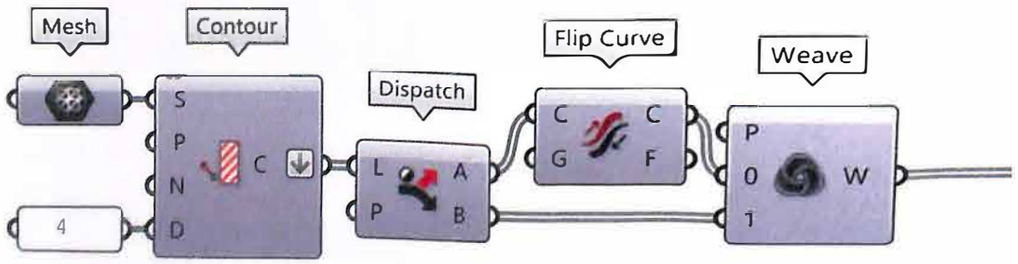


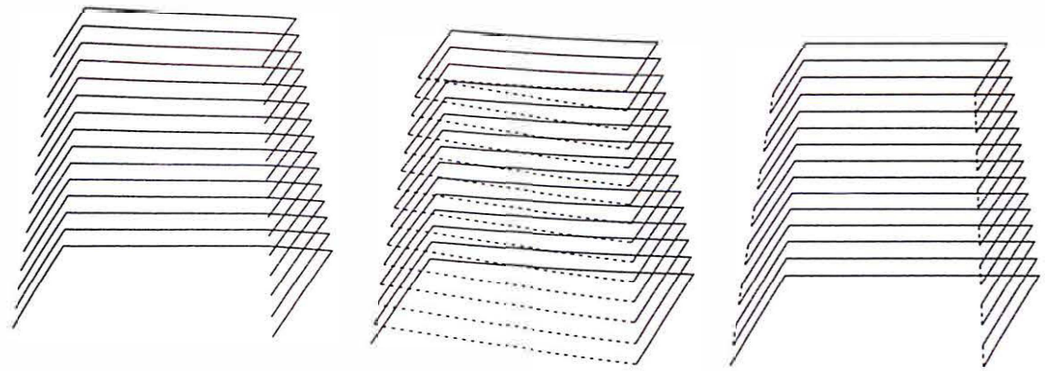


There is a big difference between an open and a closed mesh when doing contours: the end point of a polyline and the start point of the next one will be separated by a certain distance. This means that there will be a 'travel' between polylines. That is not a big issue as we can stop the extruder's extrusion during the travel, but in order to avoid unnecessary travels we could flip the direction of one out of two curves so that the start point of the next curve is close to the end point of the previous. The following steps were already explained at the *Polyline → Points → X,Y,Z* chapter, but below explains them again:

- Select one out of two curves from a flattened list with 'Dispatch'. By default, this tool uses a pattern of TRUE-FALSE. This means that the first polyline in the list will go to 'List A' output, the second to 'List B', the third to 'List A' again and so on.
- Flip either list A or B polyline's direction with 'Flip Curve'.
- Finally, we have to put them together again into one single list. For that we can use 'Weave' that will create a new list taking one item (polyline) from list 0, next from list 1, next from 0, and so on, in the same way that we wave the fingers of the two hands (if one hand would be named as 0 and the other as 1).

We can check out the resultant path as usual with 'Control points' and 'Polyline'.

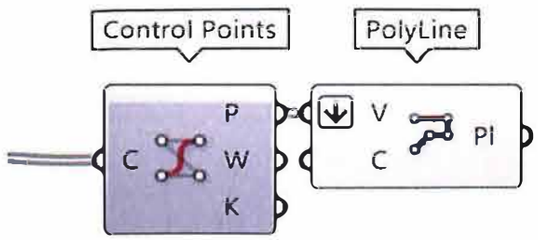




Contours

Travels (dashed lines)

Travels after 'Flip Curve'



Brep -> Mesh -> Polyline -> Points -> X,Y,Z

Brep stands for Boundary representation and is a type of 3d entity. In 3d software this term englobes any kind of surface, polysurface or extrusion that is open or closed. Rhinoceros® was developed as a N.U.R.B.S. software, so was aimed to work with surfaces and polysurfaces. We could say that a Brep is a N.U.R.B.S. type 3D object.

As usual, for 3D printing we need to transform the 3d object into XYZ coordinates. There are two options:

1. To convert the Brep into a mesh and go on as in the previous chapter or
2. To extract curves from the Brep.

In this chapter we will explain the first option. There are a few ways to transform a surface or a polysurface into a mesh. In Rhinoceros®, we use the command '_Mesh'. In Grasshopper® there is an equivalent, 'Mesh Brep', but there is also 'Mesh' component or 'Mesh Surface' (this one only works for surfaces, not polysurfaces). Getting a proper mesh from a Brep is not always so simple. Here we are not going to focus on that process, as interesting books as *AAD* by Arturo Tedeschi already explain that process in a detailed and efficient way.

But a minimum knowledge is necessary and interesting effects can be created with the different options from this tool:

Polygon Mesh Options. A single slider allows us to create a mesh with fewer polygons or more polygons in a simplified way. The 'Detailed Controls...' option opens the Polygon Mesh Detailed Options dialog box with several parameters that can be modified accordingly:

Density. it is a number from 0 to 1 that creates a mesh with more or less edges depending on how close the polygon edges are to the original surface. It is equivalent to the previous slider in the 'Polygon Mesh Options dialog box'. 0 is the slider aligned to the left and 1 the slider aligned to the right.

Maximum angle. It sets the maximum angle between the surface normal and the mesh for the mesh vertices. It will refine the mesh and make it denser where the curvature is greater.

Maximum aspect ratio. It is the proportion between the two directions of the quadrangles of the mesh. E.g. 6 means that one of the directions of the quadrangles can be no more than 6 times larger than the other. 1 means that the quadrangles will tend to be quadrangular. This option is very useful to create homogenous meshes.

This option could be combined with the density value or with the minimum initial grid quads to control the amount of faces on the mesh.

Minimum edge length. It is a number in the current unit system that controls the minimum length of the edges of the mesh. It is useful when very small edges are created, in order to get a faster and simpler mesh.

Maximum edge length. The zero value by default turns off this option. When another number is provided, it will control the maximum allowable length for the edges of the mesh in the current unit system.

Maximum distance, edge to surface. It controls the maximum allowable distance between the original NURBS and the subsequent mesh that is going to be created. It is measured from the midpoint of the edges to the NURBS surface. It is also known as 'Tolerance' in the export option as *.stl file (for 3D printing) and is very useful to get an accurate mesh from a NURBS. Zero value turns off the option. Important to note: very small values can turn in very dense meshes that could make the computer crash.

Minimum initial grid quads. It is the initial number of quadrangles of the mesh. As usual, zero value turns it off. This option can be combined with the previous ones but it is possible that if the amount of faces in this option is higher than the mesh received from previous values, it could override them creating a mesh that does not attend to previous values.

Refine mesh. It is a recursive or subdivision process on the parts of the mesh that need to be refined to meet the parameters set at 'Maximum angle', 'Minimum edge length', 'Maximum edge length' and 'Maximum distance edge to surface'. If the mesh is not refined, will be less accurate, but lighter. Its effect is very clear in combination with 'Maximum angle' option.

Jagged seams. If we are working on a polysurface, all surfaces will mesh independently. That means that the edges of the meshes of each surface do not join (stich) with the edges of the adjacent mesh. This occurs if 'jagged seams' is on. If it is not on, then the vertices and edges will match creating a continuous topology of the mesh called, a *watertight mesh*. It is better for further operations, as well as for rendering, as it does not create cracks between meshes.

Simple planes. This option will create the minimum amount of quadrangles on planar surfaces. This will ignore all settings defined on previous options except for 'jagged seams'.

Pack Textures. When polysurfaces are meshes, the coordinates for texture are also created. Pack texture applies the texture over all faces in the polysurface.

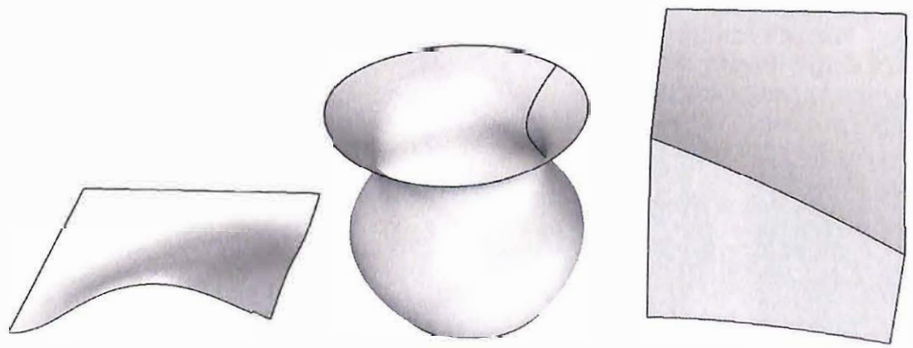
Preview. Click it to refresh the mesh displays after any change on the settings.

Simple Controls. It opens the Polygon Mesh Options dialog box.

Important to note: 'extrusion objects' do not behave as proper surfaces or polysurfaces so we have to explode and/or join them.

For further details please check the '**Help**' button on Polygon Mesh Detailed Options.

Some meshing examples with two surfaces and a polysurface of 100 mm. side length. Once we have created the desired mesh, the path can be created as in the previous chapter.



Maximum angle. It sets the maximum angle between the surface normal and the mesh for the mesh vertices. It will refine the mesh and make it denser where the curvature is greater.

Maximum aspect ratio. It is the proportion between the two directions of the quadrangles of the mesh. E.g. 6 means that one of the directions of the quadrangles can be no more than 6 times larger than the other. 1 means that the quadrangles will tend to be quadrangular. This option is very useful to create homogenous meshes.

This option could be combined with the density value or with the minimum initial grid quads to control the amount of faces on the mesh.

Minimum edge length. It is a number in the current unit system that controls the minimum length of the edges of the mesh. It is useful when very small edges are created, in order to get a faster and simpler mesh.

Maximum edge length. The zero value by default turns off this option. When another number is provided, it will control the maximum allowable length for the edges of the mesh in the current unit system.

Maximum distance, edge to surface. It controls the maximum allowable distance between the original NURBS and the subsequent mesh that is going to be created. It is measured from the midpoint of the edges to the NURBS surface. It is also known as 'Tolerance' in the export option as *.stl file (for 3D printing) and is very useful to get an accurate mesh from a NURBS. Zero value turns off the option. Important to note: very small values can turn in very dense meshes that could make the computer crash.

Minimum initial grid quads. It is the initial number of quadrangles of the mesh. As usual, zero value turns it off. This option can be combined with the previous ones but it is possible that if the amount of faces in this option is higher than the mesh received from previous values, it could override them creating a mesh that does not attend to previous values.

Refine mesh. It is a recursive or subdivision process on the parts of the mesh that need to be refined to meet the parameters set at 'Maximum angle', 'Minimum edge length', 'Maximum edge length' and 'Maximum distance edge to surface'. If the mesh is not refined, will be less accurate, but lighter. Its effect is very clear in combination with 'Maximum angle' option.

Jagged seams. If we are working on a polysurface, all surfaces will mesh independently. That means that the edges of the meshes of each surface do not join (stich) with the edges of the adjacent mesh. This occurs if 'jagged seams' is on. If it is not on, then the vertices and edges will match creating a continuous topology of the mesh called, a *watertight mesh*. It is better for further operations, as well as for rendering, as it does not create cracks between meshes.

Simple planes. This option will create the minimum amount of quadrangles on planar surfaces. This will ignore all settings defined on previous options except for 'jagged seams'.

Pack Textures. When polysurfaces are meshes, the coordinates for texture are also created. Pack texture applies the texture over all faces in the polysurface.

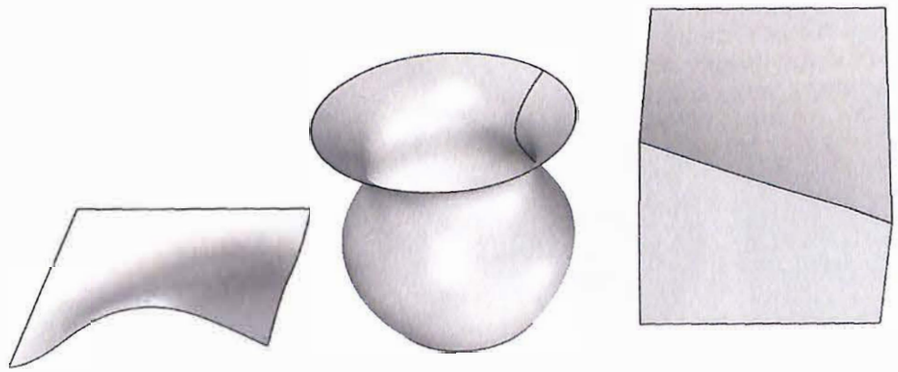
Preview. Click it to refresh the mesh displays after any change on the settings.

Simple Controls. It opens the Polygon Mesh Options dialog box.

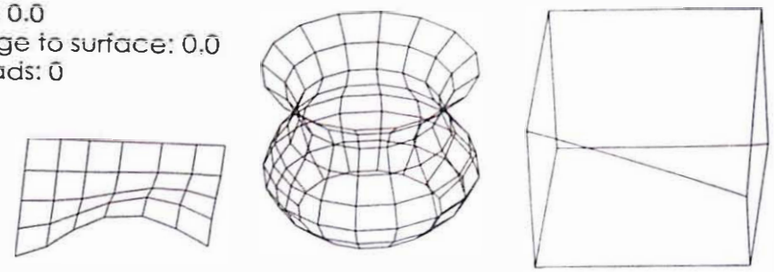
Important to note: 'extrusion objects' do not behave as proper surfaces or polysurfaces so we have to explode and/or join them.

For further details please check the '**Help**' button on Polygon Mesh Detailed Options.

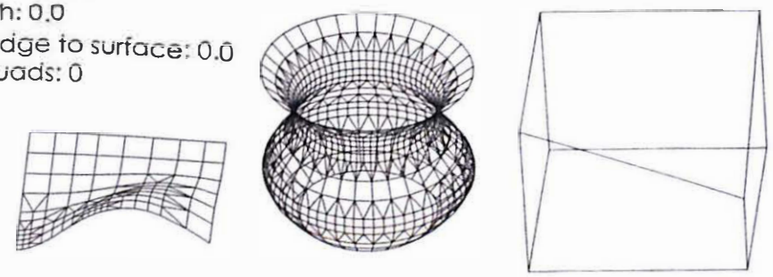
Some meshing examples with two surfaces and a polysurface of 100 mm. side length. Once we have created the desired mesh, the path can be created as in the previous chapter.



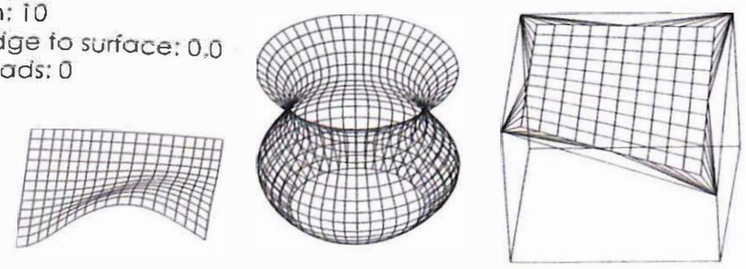
Density: 0.0
Maximum angle: 0
Maximum aspect ratio: 6
Minimum edge length: 0.0001
Maximum edge length: 0.0
Maximum distance, edge to surface: 0.0
Minimum initial grid quads: 0
☒ Refine mesh
☐ Jagged seams
☒ Simple planes
☒ Pack textures



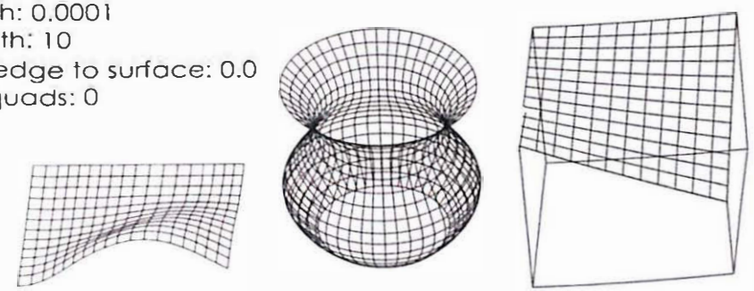
Density: 0.0
Maximum angle: 20
Maximum aspect ratio: 1
Minimum edge length: 0.0001
Maximum edge length: 0.0
Maximum distance, edge to surface: 0.0
Minimum initial grid quads: 0
☒ Refine mesh
☐ Jagged seams
☒ Simple planes
☒ Pack textures



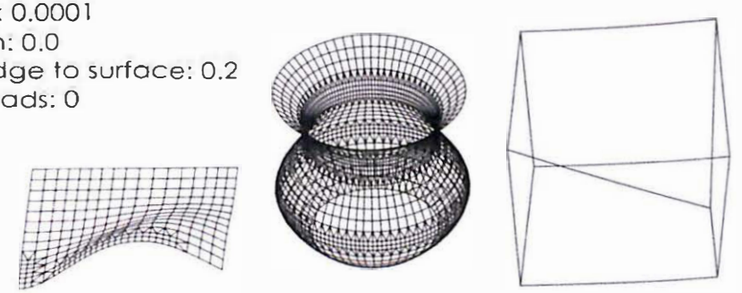
Density: 0.0
Maximum angle: 0
Maximum aspect ratio: 1
Minimum edge length: 0.0001
Maximum edge length: 10
Maximum distance, edge to surface: 0.0
Minimum initial grid quads: 0
☒ Refine mesh
☐ Jagged seams
☒ Simple planes
☒ Pack textures



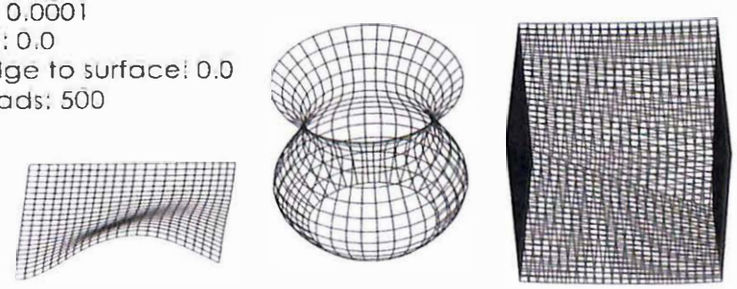
Density: 0.0
Maximum angle: 0
Maximum aspect ratio: 1
Minimum edge length: 0.0001
Maximum edge length: 10
Maximum distance, edge to surface: 0.0
Minimum initial grid quads: 0
☒ Refine mesh
☒ Jagged seams
☒ Simple planes
☒ Pack textures



Density: 0.0
Maximum angle: 0
Maximum aspect ratio: 1
Minimum edge length: 0.0001
Maximum edge length: 0.0
Maximum distance, edge to surface: 0.2
Minimum initial grid quads: 0
☒ Refine mesh
☐ Jagged seams
☒ Simple planes
☒ Pack textures



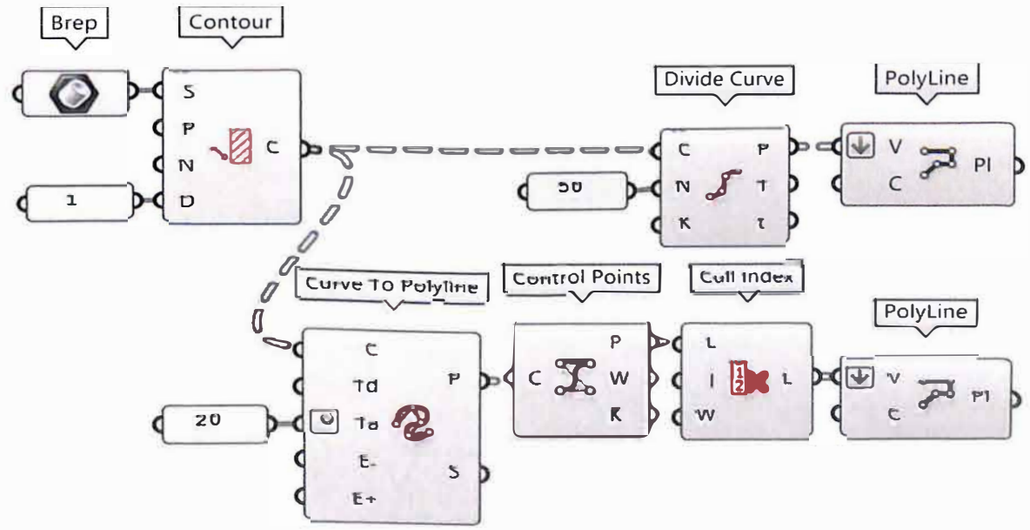
Density: 0.0
Maximum angle: 0
Maximum aspect ratio: 6
Minimum edge length: 0.0001
Maximum edge length: 0.0
Maximum distance, edge to surface: 0.0
Minimum initial grid quads: 500
☒ Refine mesh
☐ Jagged seams
☐ Simple planes
☒ Pack textures

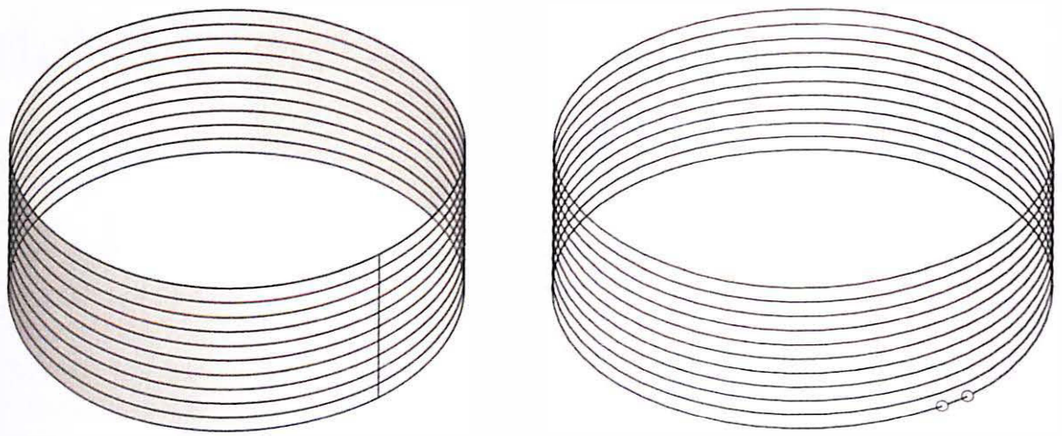


Brep -> Curves -> Polyline -> Points -> X,Y,Z

The same process followed with meshes in chapter *Mesh → Polyline → Points → X,Y,X*, can be done directly with Breps and 'Contour'. It will section the object in the same way than a 'slicing' software does. If the Brep is continuous, the resultant curves from 'Contour' will be closed. As usual, we will need to transform these curves into X,Y,Z coordinates so we can follow the steps at the *Curves → Points → XYZ* chapter, as well as dividing the curves in points, or transforming them into polylines with 'Curvetopolyline' and go on as in *Polylines → Points → XYZ*.

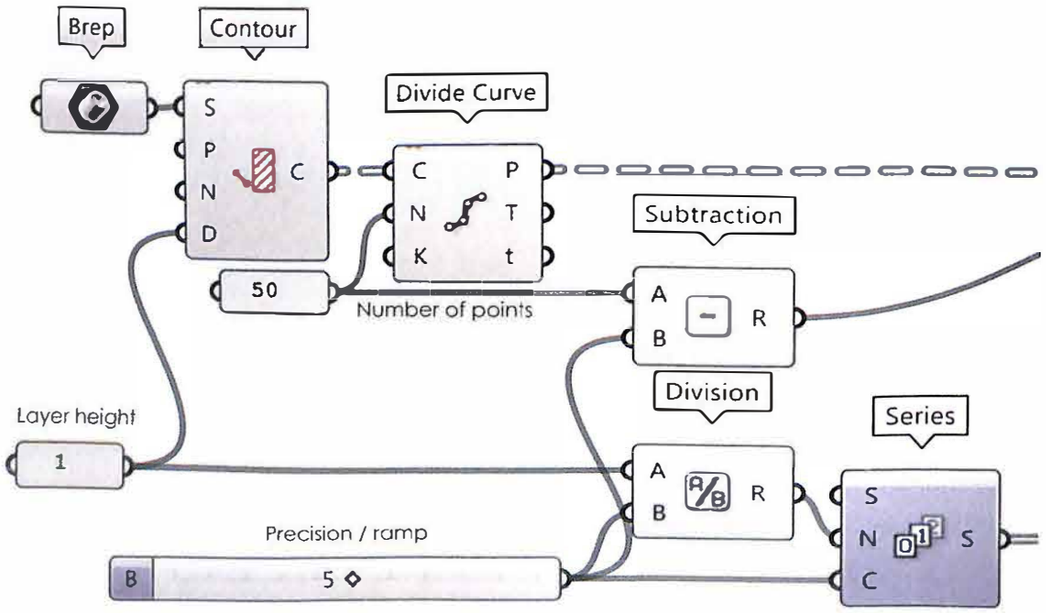
As usual, it could be interesting to create a continuous path to improve the result, as explained in the *Mesh → Polylines → Points → XYZ* chapter. Remember that one continuous curve means that the extruder will not stop extruding. This way we will avoid the seam between layers, the travels between end points and the retraction in the g-code. This chapter will mainly focus on achieving this goal. These were the two methods explained, but this time using a Brep instead of a Mesh as input:

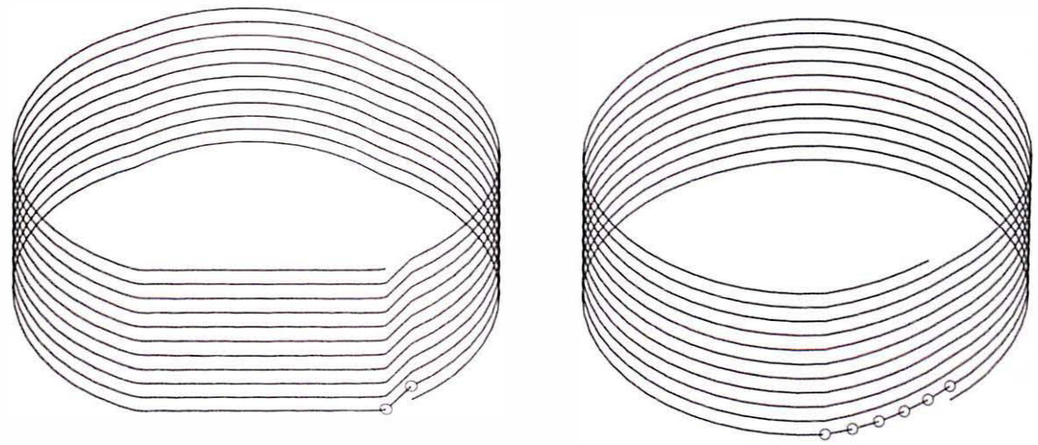




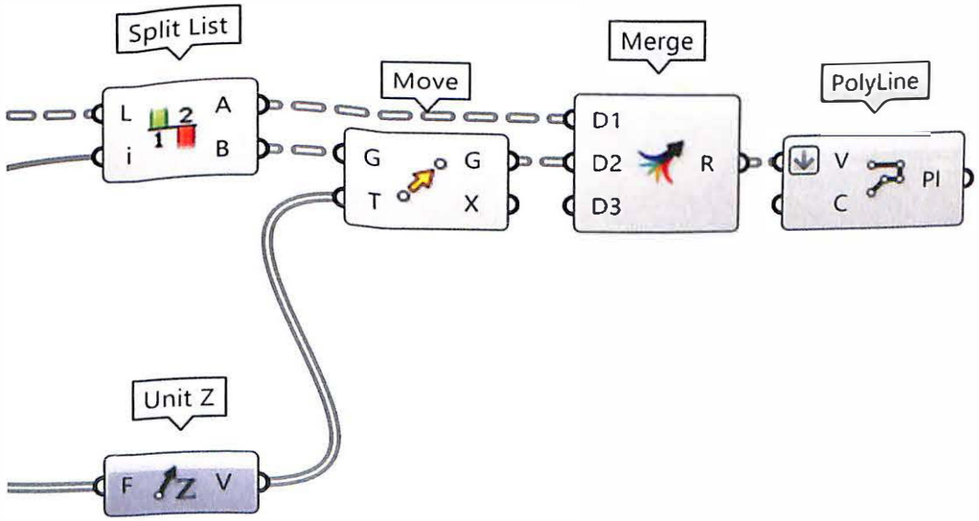
A smoother transition or 'ramp' between layers can be created. But as usual, the final result will depend on our geometry.

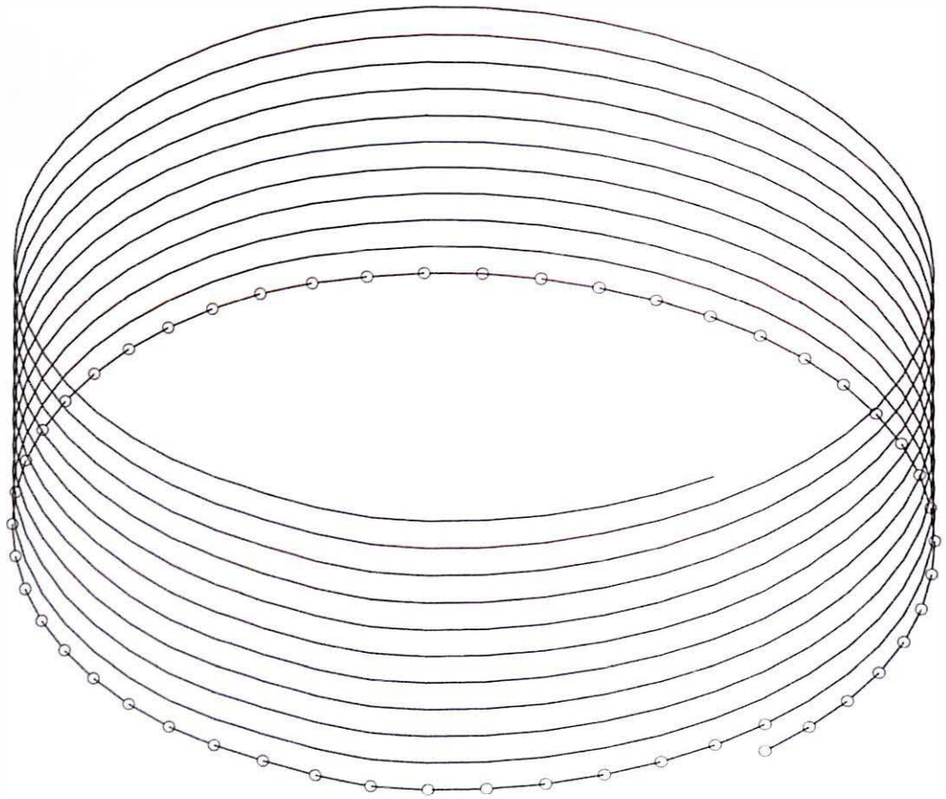
We can select several points of the curve (e.g. the last 5) and move them progressively to the next level curve (or height). First, 'Split' the list of points and work with the last 5 points moving them in the Z axis to the correspondent average height. If our layer is 1mm. height, divide it by 5 so the first point will move 0.2mm, the next 0.4 mm. etc.

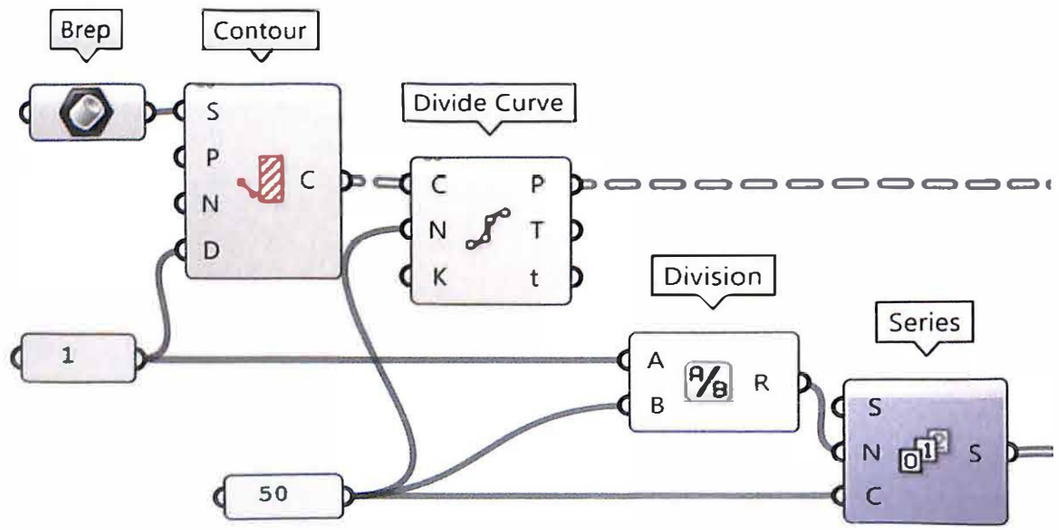


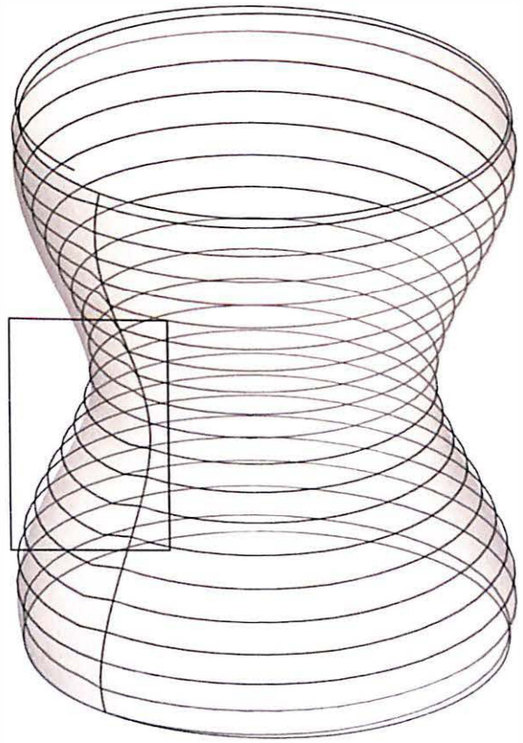


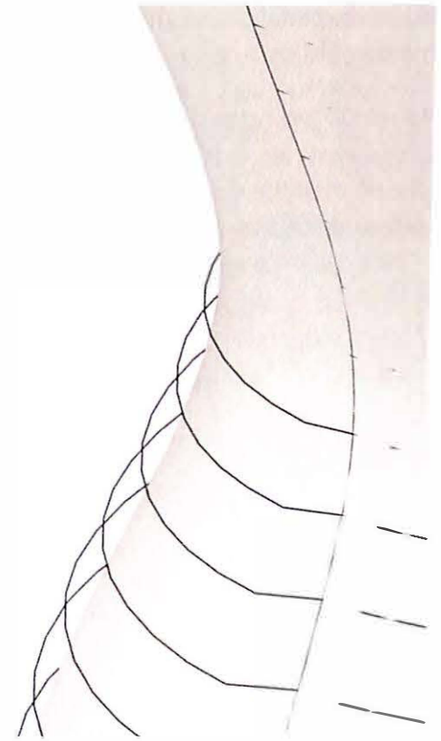
Instead of only 5 or a few, all the division points could be selected to make a continuous ramp. That makes the definition even more simple and the curve smoother. In a cylinder like the example, the result will be a spring. On the contrary, the touch with the build plate is not as good as in the previous examples, as the curve starts to raise from the first point.



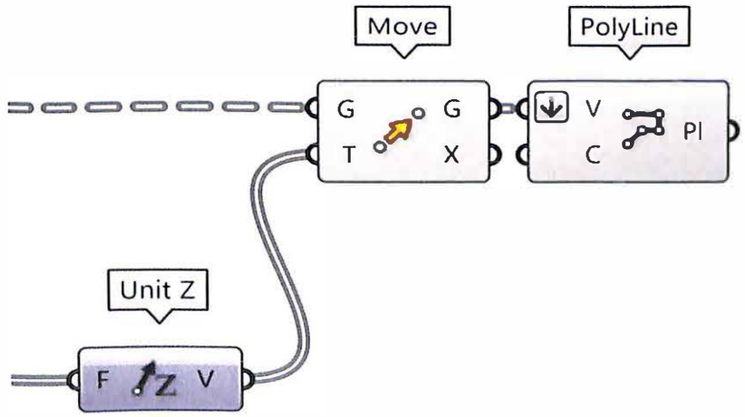






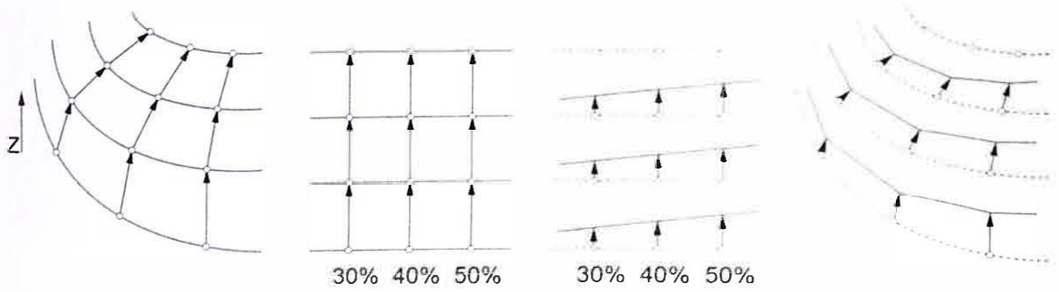


Depending on the geometry of the object, this strategy could fail. When the Brep has no vertical walls, we realise that the start and end points of consecutive contours do not perfectly match:



That happends because we are moving the points along the Z axis. The correct movement should be carried out in the direction of the next contour.

As usual, we have to work further to define the most suitable direction for the movement of each point. The strategy now is to move every point in the direction of the equivalent point in the next curve by a certain percentage that is determined by the amount of points selected to make the ramp.



For each point's movement we need to create a vector and multiply it by a percentage. A simple way to create a vector is using 'Vector 2Pt'.

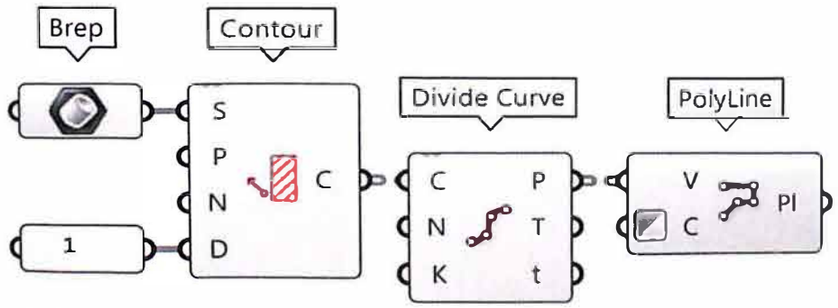
There are several strategies to find the adequate points. An easy, geometric one, lies on drawing a polyline that connects the first point of a curve with the first of the next and so on. This is a typical exercise on Grasshopper®, solved using the component 'Flip Matrix'. This tool creates the opposite data structure to the current one.

See example of a frustum for a better understanding of 'Flip Matrix':

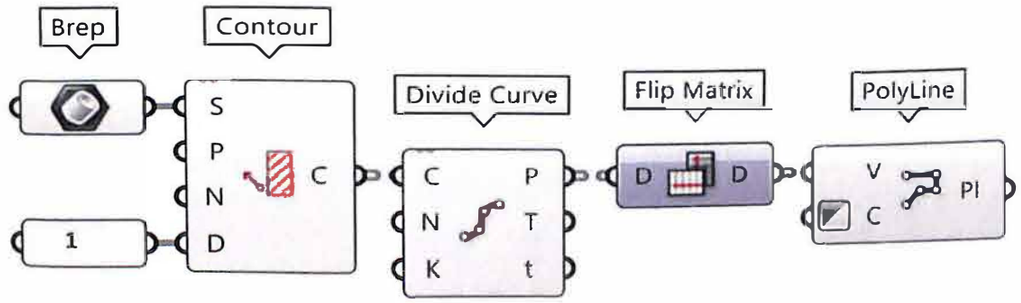
(1) (2)

For the movement we need to use vectors (picture in the next page). Use 'End points' to create the vectors with 'Vector 2Pt'. Explode the polylines into their segments with 'Explode' component. Define a percentage between '0 to 1'. Use 'Range' to divide a domain from '0 to 1' by the amount of points. The output has one more value than what is needed.

E.g. if we have a total of 5 points and we ask 'Range' to divide a domain from 0 to 1 in 5 parts, the resultant list will be as following: 0.00, 0.20, 0.40, 0.60, 0.80, 1.00. A total of 6 values for 5 division points. It is completely logical. But that extra value will have a non-desirable effect on our 5 points list. To make them fit, it is better to ask to 'Range' to divide by one less unit. Add 'x-1' expression to the 'Steps' input by right clicking on it.



(1)



(2)

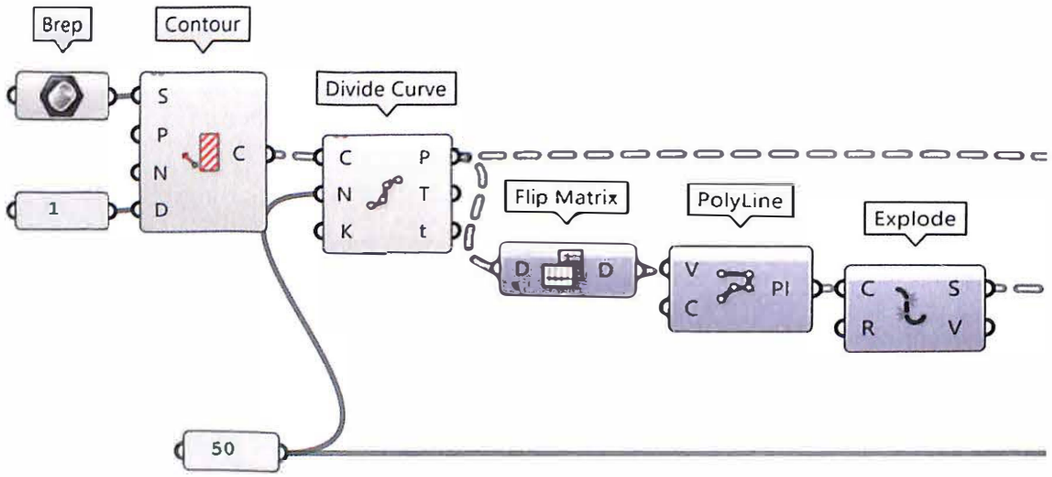
We are now dividing the Domain in **5 - 1 = 4** steps, outputting 5 values:

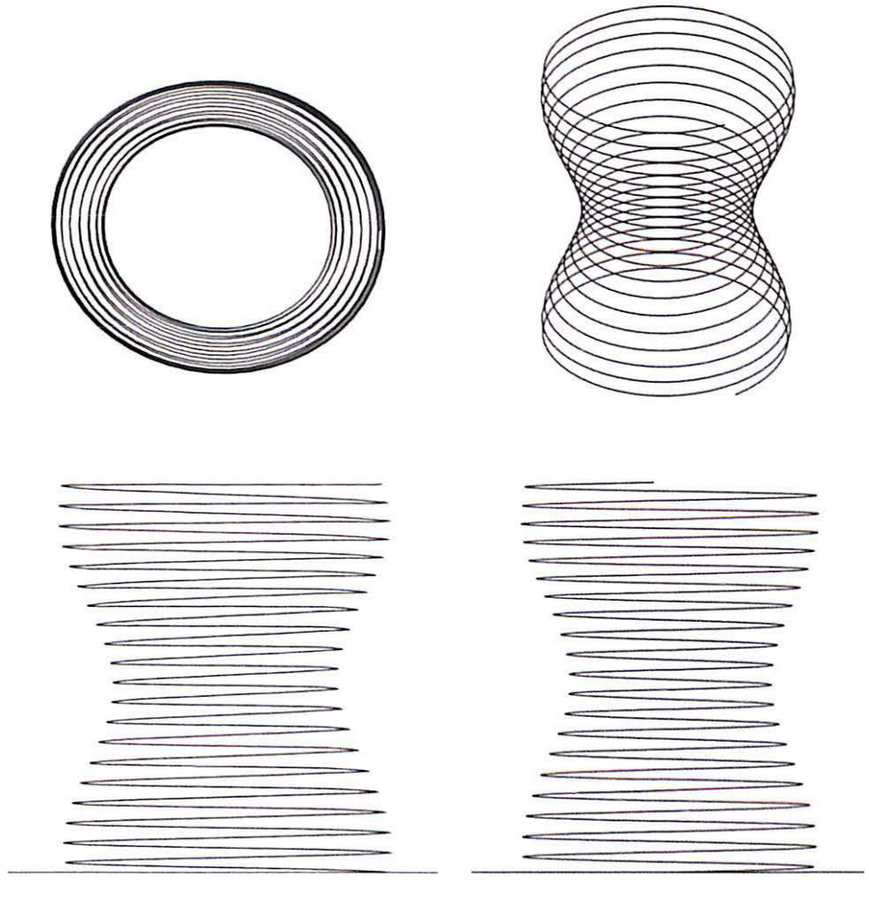
0.00, 0.25, 0.50, 0.75, 1.00

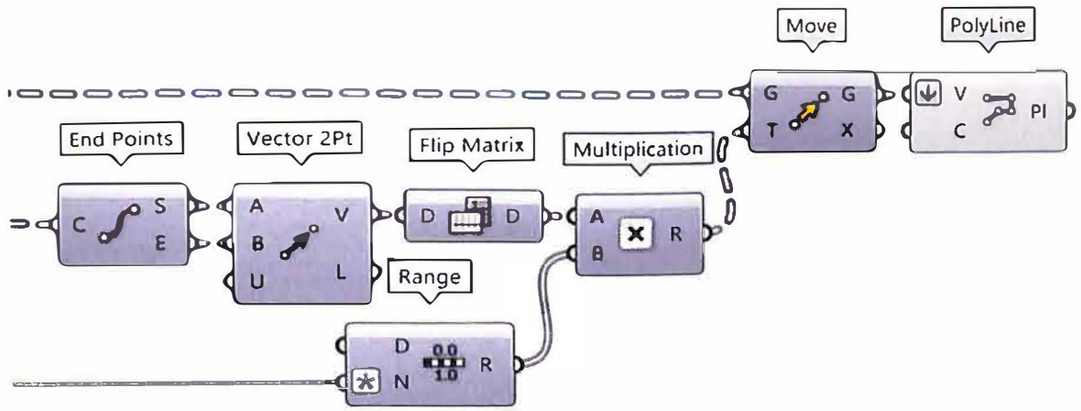
So, no matter the number of points we have (50 in the example below), the list of divisions of the domain will fit.

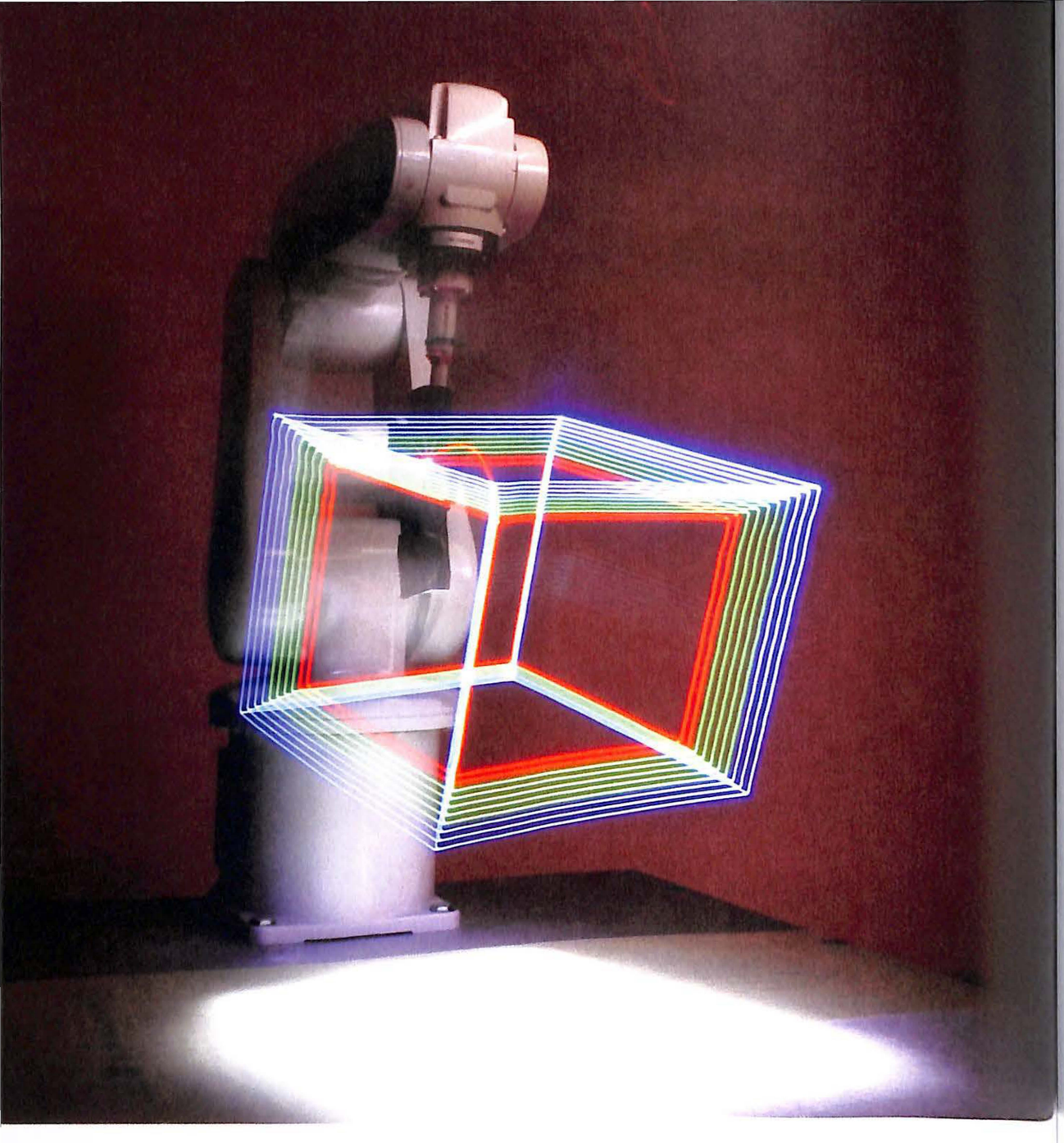
Before the final motion of the points along the vectors, we need to go back to the previous data structure. One more 'Flip Matrix' component will revert the effects of the first one.

This definition works with points from curves; these can be contours like in the example, or any other ordered distribution of curves, so it could be applied to any similar distribution of points generated by polylines, curves, meshes or Breps.









Light painting experiment with an RGB led and an ABB IRB120 robot arm at UEM (Madrid).

nD printing: robot arms

The purpose of the book is not to control robot arms for 3D printing as it is much more complex than a 3D printer, but we want to provide a slight introduction to this topic.

The use of robot arms is widely spread across industries. The most common uses are painting, palletizing and pick&place operations. However, a clay or filament extruder could be attached to the robot's flange. The extruder of a standard 3D printer is normal (perpendicular) to the build plate's plane and it cannot rotate or modify its angle. So, it is easy to control its movement via points with 3 coordinates as explained in previous chapters. When printing with a robot arm more variables need to be taken in account. An industrial robot arm, commonly features six axes, also known as degrees of freedom. It means that if we use an extruder as an end-effector, the tip does not have to be normal to the World XY plane anymore. For the same coordinate X,Y,Z, the extruder's tip could be normal to an infinite number of planes. So, one of the main differences between 3 axes 3D printing and 6 is that instead of creating our code by X,Y,Z coordinates we have to provide planes. A plane in Grasshopper® is defined by its origin and its normal vector:

World XY plane: O(0,0,0) Z(0,0,1)

World XZ plane: O(0,0,0) Z(0,1,0)

World YZ plane: O(0,0,0) Z(1,0,0)

There are several menus and toolbars to deal with planes in Rhinoceros® and Grasshopper®.

Apart from the geometric differences, there is another surprising fact with robot arms. There is no standard coding language used as what it could be g-code. Each brand has its own language:

ABB → RAPID (Robot Application Programming Interface Dialog)

KUKA → KRL (KUKA Robot Language)

FANUC → KAREL (Named after Karel Capek, who introduced the word 'robot')

Comparing G-code with RAPID, there is a remarkable difference:

G-CODE → G1 F1800 X10 Y10 Z10 E5

RAPID → MoveL [[10,10,10],[-0.5,0.0,1,0.0]],Speed100,Z10,Tool1

[10,10,10] is the origin point of the plane

[-0.5,0.0,1,0.0] are the quaternions for the orientation

Speed100 is the speed for the movement

Z10 is the zone or precision

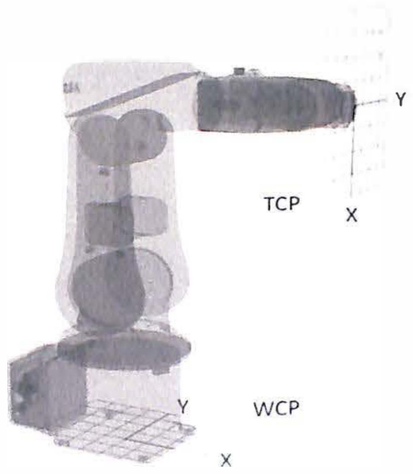
Tool1 is the declared tool.

The position and orientation of the robot flange can be described by quaternions based on Euler's angles. Euler's angles can describe the movement in space of a rigid solid through 3 rotations:

1st: along Z axis with a domain from 0 to 2Pi (360 degrees)

2nd: along X axis with a domain from 0 to Pi (180 degrees)

3rd: along Z axis with a domain from 0 to 2Pi (360 degrees)



Unlike 3D printers that have one coordinate system (that could be at the centre point on the build plate or in one of its corners), a robot arm has two coordinate systems: one for the movement of the body and another for the flange. The first is known as World Coordinate Plane (WCP) which is fixed, and the second is the Tool Coordinate Plane (TCP) which moves with the flange.

Controlling all these parameters and the robot arm's language, a new world of possibilities opens for 3D printing.

With a curve or path in 3D we can chose the robot's flange plane to follow it in as an XY plane or as a tangent plane with the tool perpendicular to the surface as in the pictures below. That makes a big difference with 3D printers.