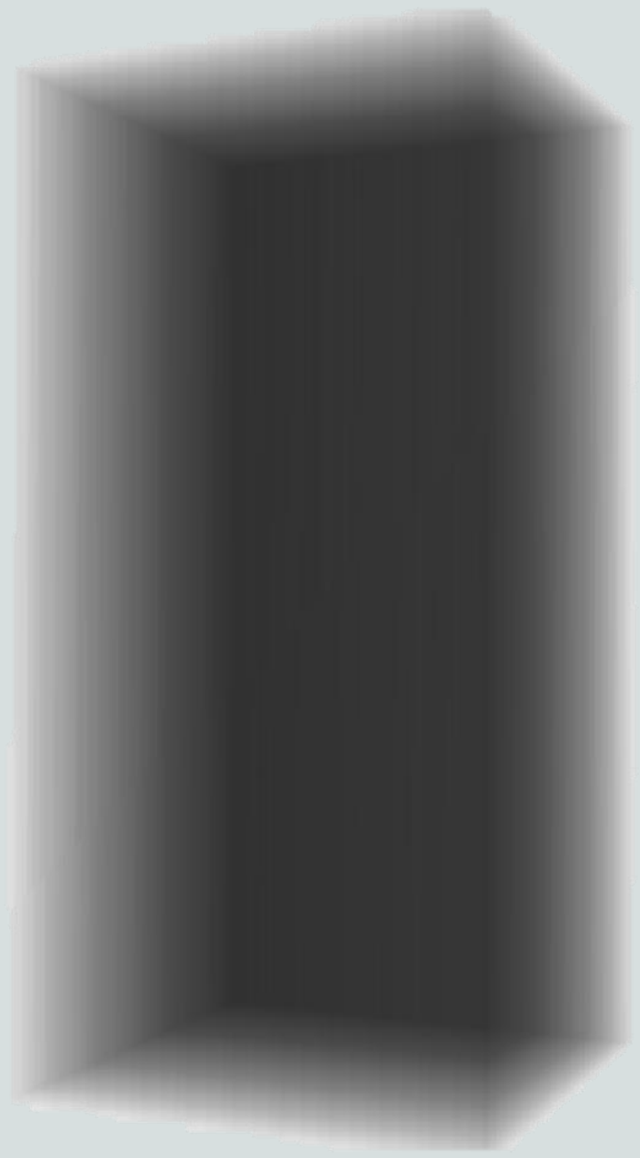


Autodesk : project monolith

Monolith is a volume modelling software, a hybrid between a 3d CAD and a 3d image editing application. The user creates models by manipulating three dimensional images made of voxels. Monolith has been developed with multi material 3d printing in mind. Hence each voxel is made of two values, one of them is used to define the boundary between solid and void and the other the mixing ratio between two materials. The mixing ratio is realized using traditional typographic rasterization techniques extended to three dimensions. Therefore the user can define rasterization patterns with variable degrees of optical and elastic anisotropy.

1 Terminology

Voxel Image: A voxel image is a three dimensional structure that consists of one or more voxel channels. It is the equivalent of a regular 2d image. 2d Images usually consist of three colour channels [red green and blue]. Voxel images in monolith can have a variety of channels. The most common are the Shape channel that defines the boundaries between solid and void, and the material ratio channel that defines the mixing of two materials within the solid region. In addition there are mask channels used for compositing voxel images and specialized channels which hold for example structural analysis results when using the analysis tools. All channels within a Voxel image have the exact same resolution [number of voxels along the x, y and z axes].

Voxel channel: A voxel channel is basically a three dimensional array of numbers representing the variation of some scalar quantity over the voxel image box. In general most channels are normalized to the 0.0 to 1.0 or the -1.0 to 1.0.

Layer: Layers are the building blocks of a monolith model. They are built as a stack with the top layers composited on the lower ones. For example a layer may create a box like field, another layer on top of it may smooth it, and another might apply a twist modifier. In that sense layers may be seen as voxel images, or image filters that are superimposed. The stack of layers is hierarchical as there are layers that can have child layers within them. For example the transform Layer manages an independent stack of layers that acts as a group and allows you to move scale and rotate the resulting object independently from the overall image.

Shape channel: Every monolith image has a shape channel. Although the shape channel itself is “fuzzy” in that it contains a gradient of values [usually from 0 to 1] that vary continuously throughout the image volume, the purpose of this channel is ultimately to define a shape with a hard boundary. You can think of the shape channel as a grayscale 3d image that defines a gradient field. However at some point when the field is visualized or printed, it becomes binarized by applying a threshold [the isovalues]. Regions within the threshold are considered solid and regions outside the threshold are considered void.

Material ratio channel: The material ratio channel is intended as a helper for dual material printing. It describers the material mixing ratio at each point in space. A value of 0.0 would mean that the first material is applied at the corresponding voxel and a value of 1.0 would correspond to the 2nd material. A value of 0.5 would be interpreted as a 50% mixture of the two materials, therefore mixing their colour, elastic and optical properties.

Isovalues: An isovalue is a number that represents a threshold in a field. It is used in order to define surface contours in a three dimensional field [called isosurfaces]. Its primary purpose in monolith is to define the contour level at which the shape field transitions from void to solid.

Isosurface: An isosurface is the surface that crosses all voxels in a field at a particular value [called the isovalue]. It defines the boundary or contour that separates voxels of high density [d>isovalue] from voxels of lower density [d<isovalue]. Isosurfaces are explicitly calculated for two reasons. First when visualizing the cut-off boundary of the shape field to reveal the shape that would result from its printing. Second when outputting STL meshes for 3d printing.

Volume rendering: The volume rendering technique can visualize the voxel image fields without the need for explicitly calculating the isosurfaces, therefore it is much faster than isosurface rendering and is a preferred method when dealing with complex high resolution voxel images [e.g. medical datasets]. Unlike isosurface rendering it can also reveal the inner structures of transparent and translucent objects, as well as show the fine structures of the application of rasterization patterns. It is based on the rendering of planes parallel to the camera. Each plane represent s a slice through the voxel image and inherits the colour and transparency of the intersected voxels. The gradients of the shape and alpha fields are used to estimate volume normal and lighting effects.

Rasterization: Rasterization is the process of generating the bitmap slices that the 3d printer actually prints, from the continuous voxel image fields. As with 2d images there are various ways to convert a low resolution grayscale field to a high resolution black and white binary bitmap that determines the exact placement of ink or material. This is the domain of 3d typography. In monolith the rasterization techniques are described by Raster patterns.

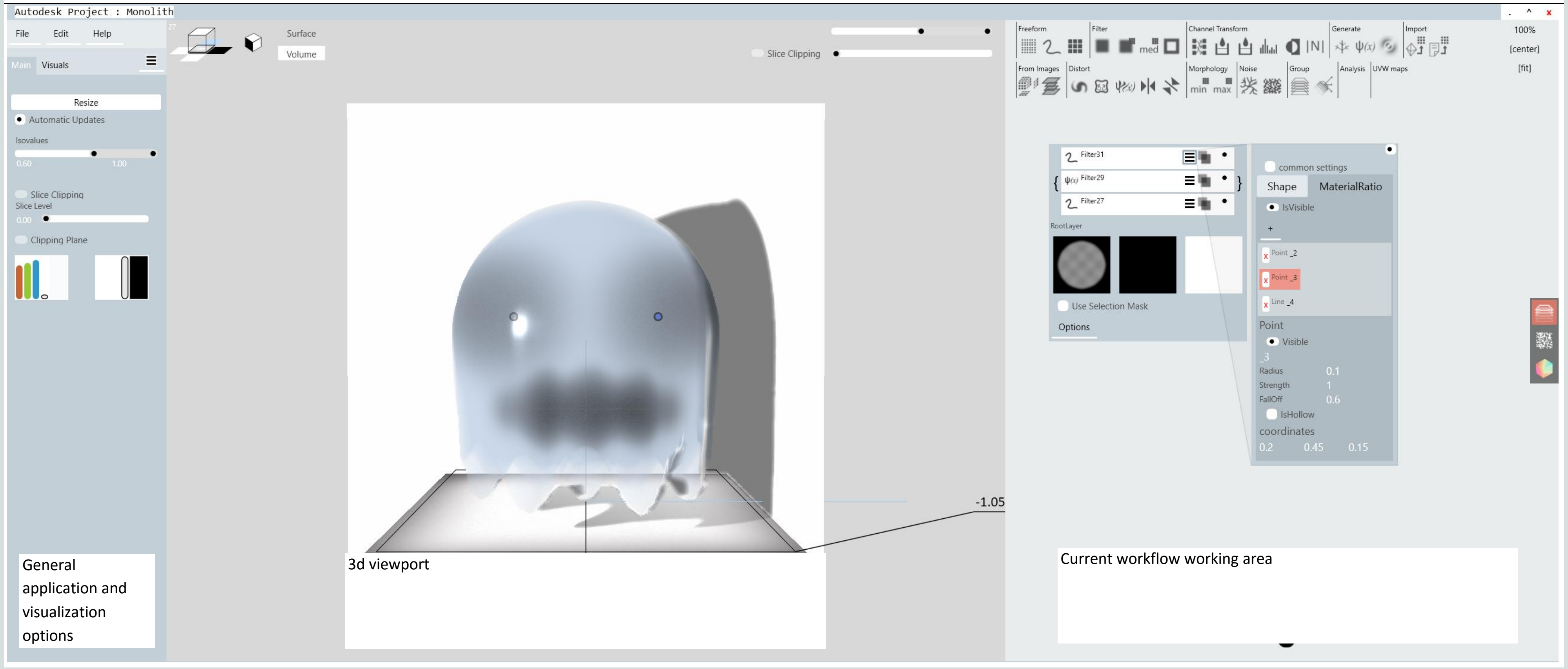
Raster pattern: A raster pattern determines how the continuous gradients in voxel images are translated into high resolution binary bitmaps. For example there are patterns that use random noise to achieve blending and others that rely on three dimensional half tone techniques. In the end the raster pattern determines the microstructure and grain of the 3d printed object. Therefore one can achieve outcomes with anisotropic elastic and optical properties.

Mask mesh: The mask mesh is used in to better control the outer boundary of the 3d printed object as well as help integrate more traditional 3d printing geometric workflows. The mask mesh can be created in any 3d modelling software as a watertight mesh surface. When imported into monolith it overrides the shape channel in the final composition step. Any voxel that falls outside the mask mesh will be void. For example you can create a shape in a surface modelling software and bring it into monolith as a mask mesh. Then you can use monolith to describe how materials vary within this shape, something that requires volumetric rather than surface modelling. The advantage of using a mask mesh is that it can capture sharp corners and discontinuities better than voxel based contours.

Lith file: A lith file is a file that is used to store a monolith project. It captures the whole layer structure and not just the final voxel image outcome. Therefore it allows a parametric re-editing of the various primitives and filters that make up a monolith project.

Vol file: The vol file is a file that contains a voxel image. It is more similar to conventional image files like tiff and png. Within the vol files the series of channels that make up an image are stored as arrays of double precision numbers. In addition the vol file stores the physical dimensions of the 3d field in inches.

2 Monolith UI



Automatic Updates

Isovalues 0.60 1.00

Slice Clipping

Slice Level 0.00

Clipping Plane

common settings

IsVisible

Point

Visible

_3

Radius	0.1
Strength	1
FallOff	0.6

IsHollow

coordinates 0.2 0.45 0.15

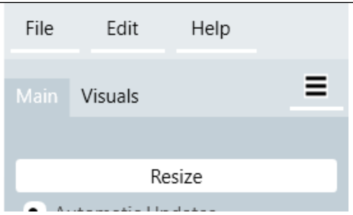
Monolith’s user interface is divided into three main regions:

a. **The left side bar**: You can use this area to access application wide settings and functionality.

b. **The Viewport**: Where you can interact with the three dimensional representation of the model.

c. **The right working area**: This region’s contents are contextual depending on the current active workflow. You can change workflow using the buttons on the far right edge of the window

2.1 General settings user interface



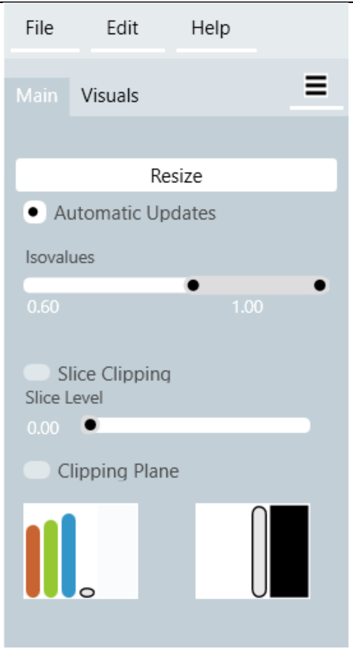
Menu

Settings: Use the tabs to switch between general application settings and visualization options.

Use the properties button ☰ in order to access less frequently used settings.

The side bar consists of the system menu at the top and a collection of panels that give access to various settings within monolith.

2.1.1 Main side-bar



Automatic Updates ●

Isovalues 0.60 1.00

Slice Clipping

Slice Level 0.00

Clipping Plane

Change dimensions and resolution of voxel image

If "automatic updates" is off the model will not update each time you make a change but rather will wait for you to press the update button manually. This is useful for high resolution or very complex models.

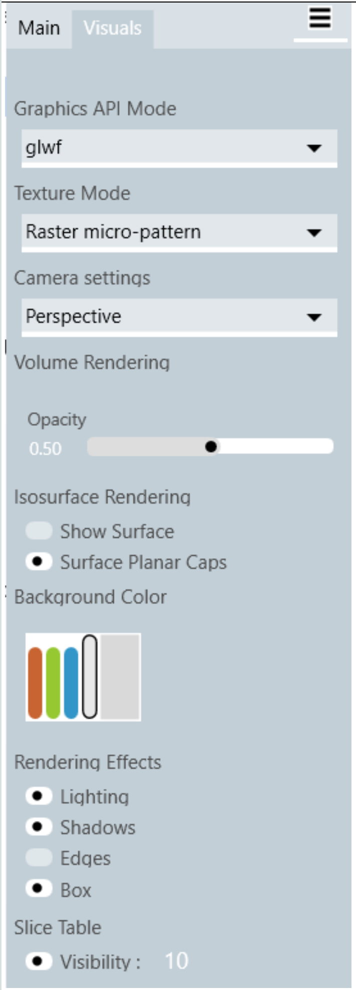
"isovalues" controls the range of the shape layer that is considered solid.

Slice clipping will make the portion of the model above the slice level invisible.

The clipping plain allows you to slice through the model at an arbitrary angle in order to inspect interior structures in cross-section.

Here you can set the colors of the two materials that are mixed along the material gradient.

2.1.2 Visuals side-bar



Graphics API Mode: glwf

Texture Mode: Raster micro-pattern

Camera settings: Perspective

Volume Rendering

Opacity 0.50

Isosurface Rendering

Show Surface

● Surface Planar Caps

Background Color

Rendering Effects

● Lighting

● Shadows

Edges

● Box

Slice Table

● Visibility : 10

Using this combo box you can pick a different rendering API for the application. Once you change this setting you will need to restart monolith for it to take effect. Depending on your graphics card different modes might give you higher performance especially when using real time volume rendering.

The texture mode determines the type of visualization that is used when in volume rendering mode.

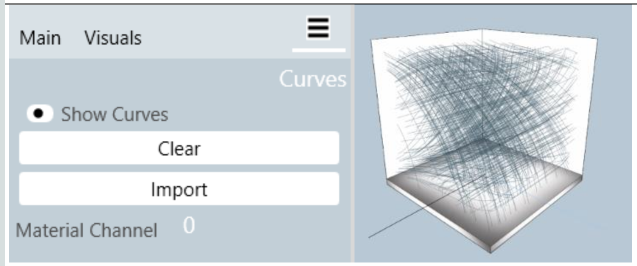
The viewport camera can switch between perspective, orthographic and axonometric view

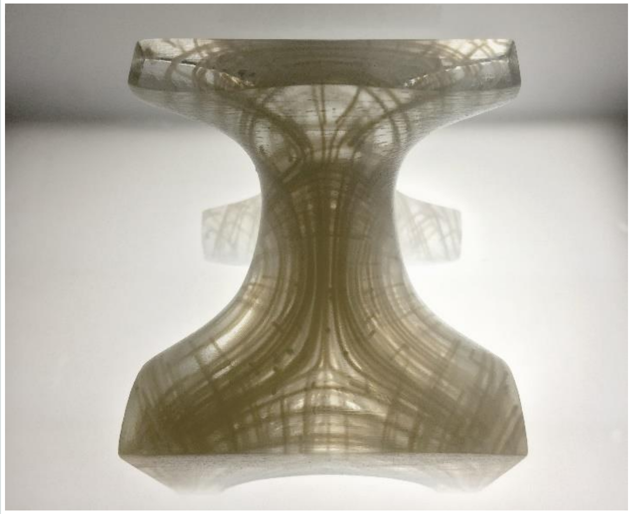
This setting determines the translucency of the model when in volume rendering mode. Lower values may help resolve internal

These settings determine whether the isosurface will be rendered and if so whether the caps that block its sides will be computed.

!!!The isosurface calculation can be slow for complex or high resolution images and it is implemented as an asynchronous operation. Therefore you may see the Isosurface checkbox turning pink. In that case the isosurface is being computed in the background and will be displayed as soon as it is ready. That means that the isosurface may not reflect the exact current state of your voxel model. [The volume rendering however always updates in sync with the voxel image, so use this rendering mode for complex operations]

2.1.3 Curves side-bar



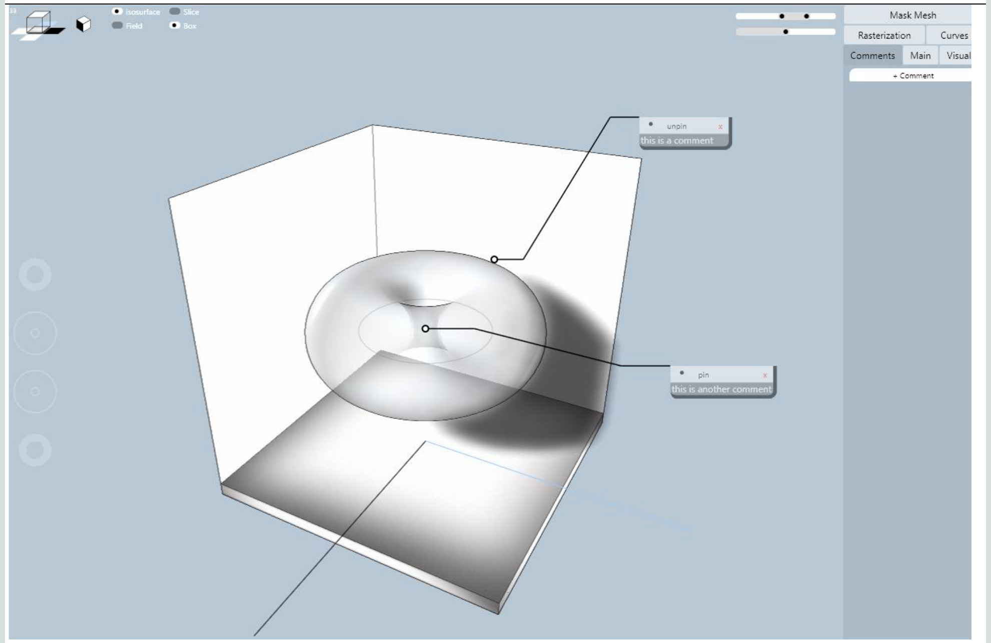


Use the curves side bar to import a set of curves into monolith. The curves are composited at the end of the rasterization process. In this way you can embed very fine detailed vector geometry in three dimensions without the need to generate heavy meshes. Therefore, you can achieve hairline printing [curves with widths equal to a single material dot at the maximum printer resolutions]. This technique is useful for embedding text and diagrams in 3d fields or visualizing streamlines.

2.1.4 Mask Mesh side-bar

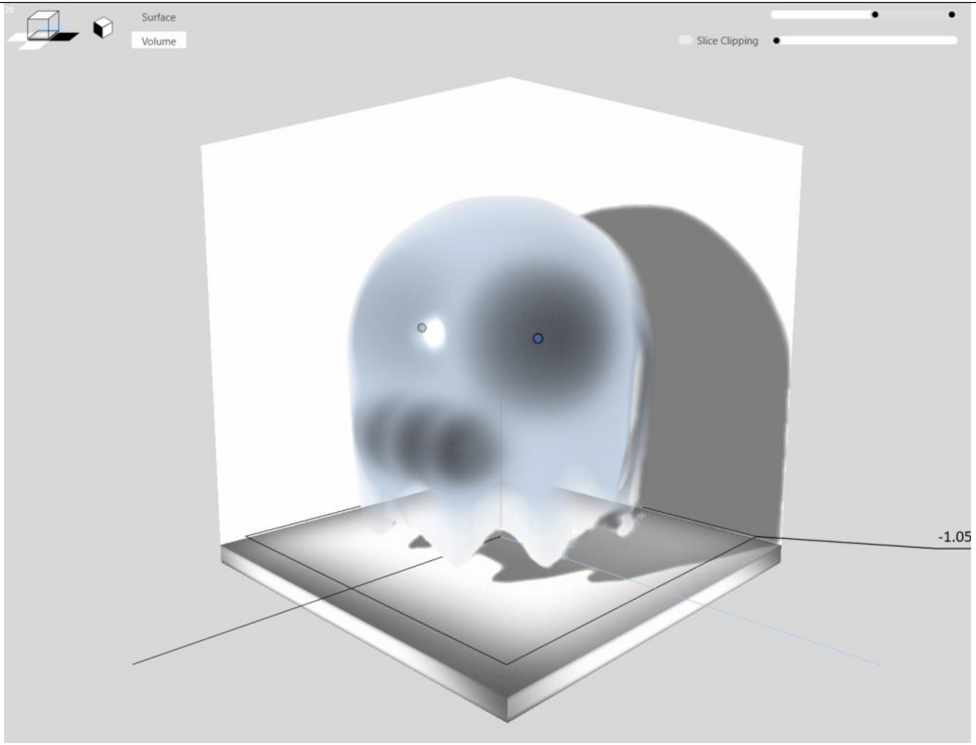
Using a mask mesh will allow you to set an outer boundary for the voxel field defined by a water tight mesh modelled in another software. This will enable the use of existing geometry to define the shape while you can still take advantage of monolith in order to define the material distribution, mixing and micro-patterning within this shape.

2.1.5 Comments side-bar



You can attach comments to any monolith model. After adding a comment you can edit its text by clicking on it. You can also drag the comment or its anchoring point in 3d. In addition comments can be pinned or unpinned to the 2d viewport overlay.

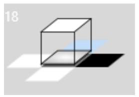
2.2 The viewport



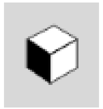
Monolith's viewport allows the visualization of the volumetric model in real-time as well as interaction with several tools and workflows in the application.

You can **pan, zoom and rotate** the camera using the left, middle and right mouse button while dragging the mouse.

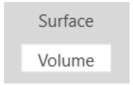
There are only a few controls available at the top of the viewport:



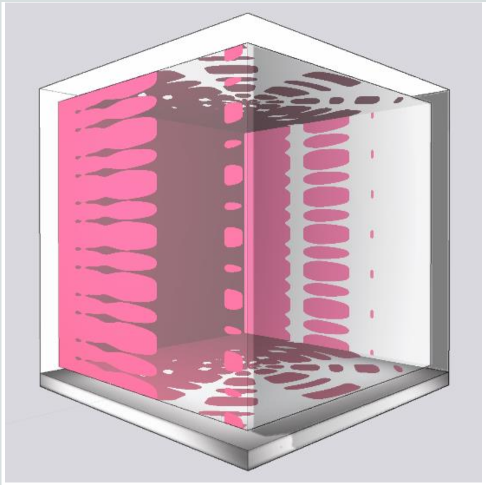
Click on any surface of the cube to set the camera to the respective side [black is the positive x axis and blue the positive y axis direction]

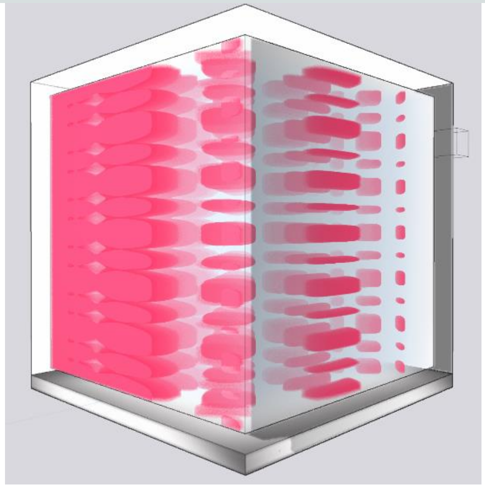


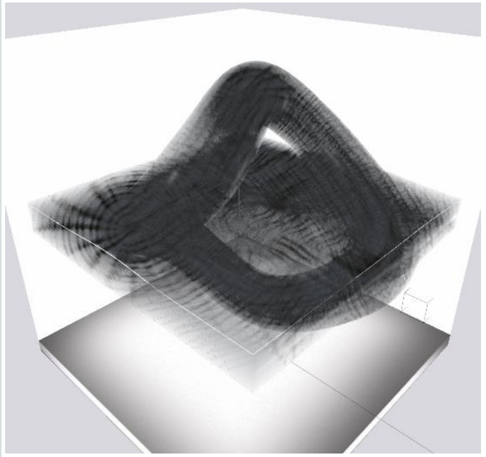
This button will **re-centre** the camera on the model



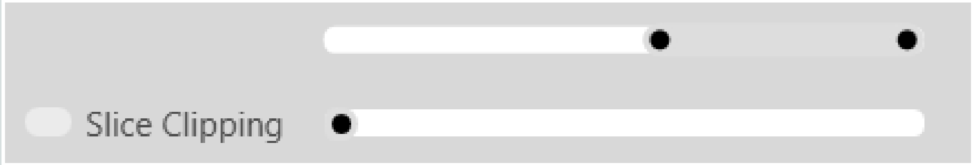
Here you can select between **surface** and **volume** rendering.





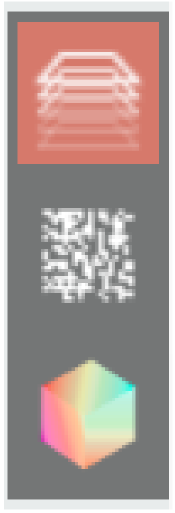


Surface rendering [top left] is faster but as you can see from the two images above it won't allow you to see the internal structures and patterns within the model. Volume rendering [top right] may be slow for some older graphics cards.



These sliders are a shortcut to the same functionality as in the main side bar.
The top slider sets the solid range threshold for the isosurface.
The bottom slider determines the level for the current horizontal section.

3 Top level workflows



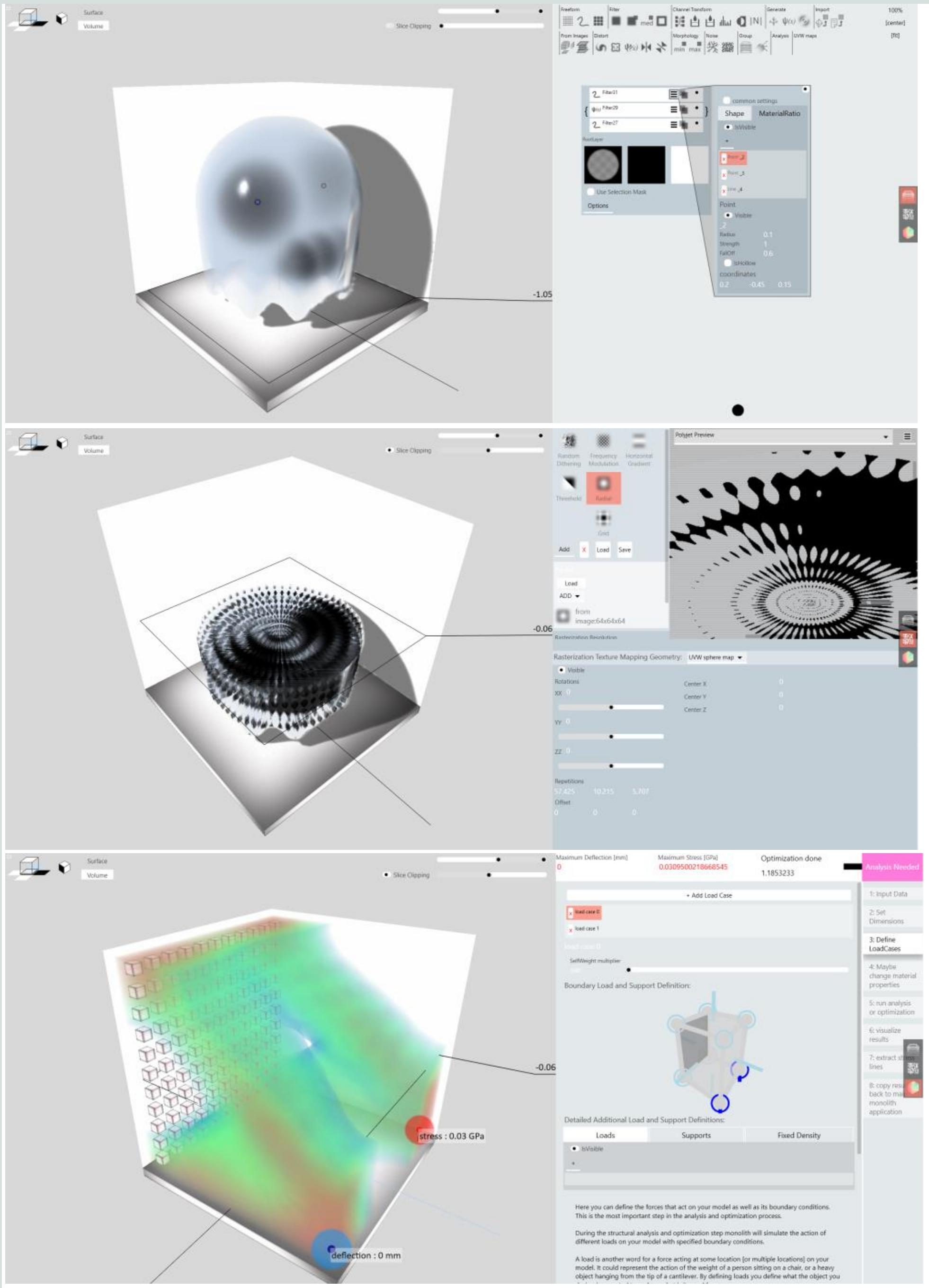
Design and layer compositing

Rasterization and micro pattern design workflow

Structural analysis workflow

The three buttons at the far right edge of the main window let you switch between top level workflows. Currently monolith has three workflows which we will refer to from now on as Design, Rasterization and Structural simulation.

Switching between workflows completely replaces the user interfaces of the working area on the right side of the window, but it does not destroy the information in the current workflow. You can switch between them freely at any time.

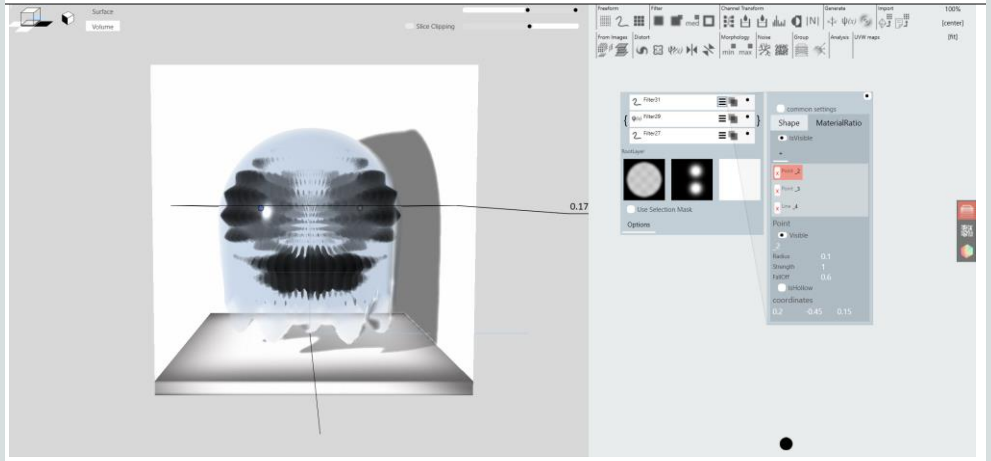


Here you can define the forces that act on your model as well as its boundary conditions. This is the most important step in the analysis and optimization process.

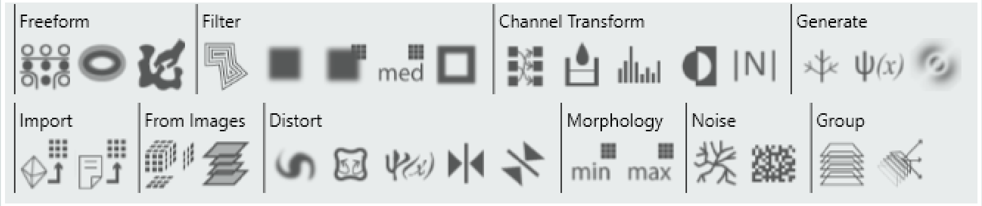
During the structural analysis and optimization step monolith will simulate the action of different loads on your model with specified boundary conditions.

A load is another word for a force acting at some location [or multiple locations] on your model. It could represent the action of the weight of a person sitting on a chair, or a heavy object hanging from the tip of a cantilever. By defining loads you define what the object you

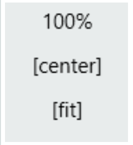
3.1 Design workflow



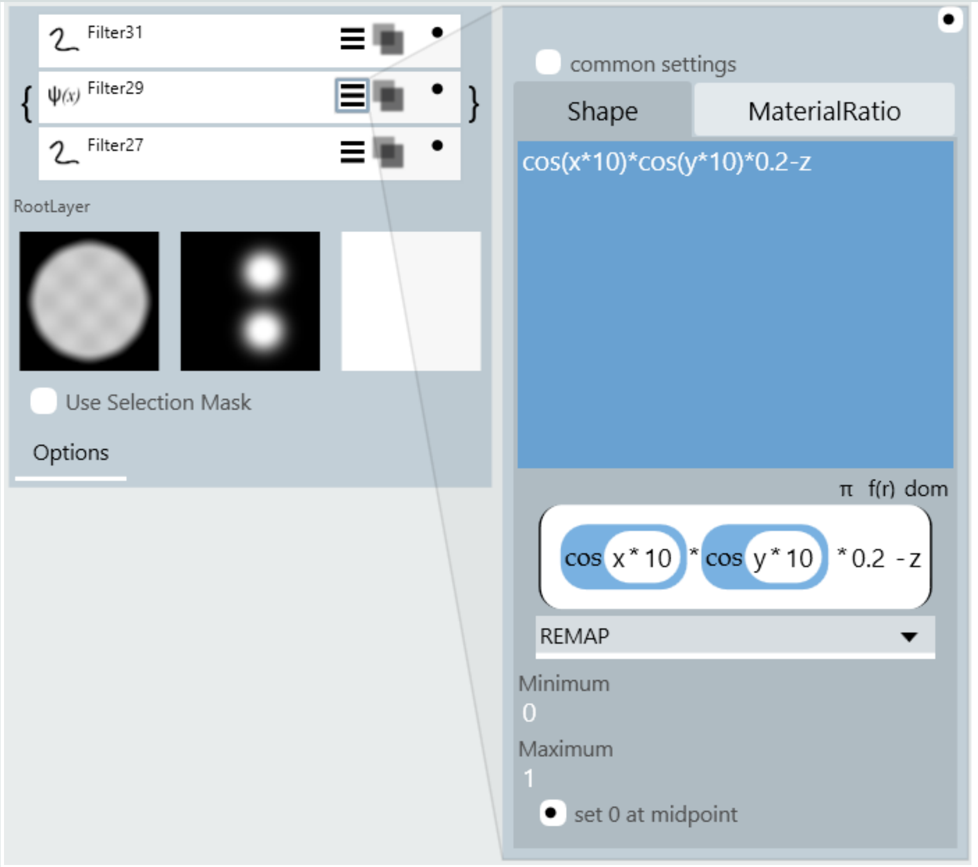
When the design workflow is active the user interface switches to the layer compositing panel [above right]. In this panel the user creates and manipulates layers that contain three dimensional objects or effects.



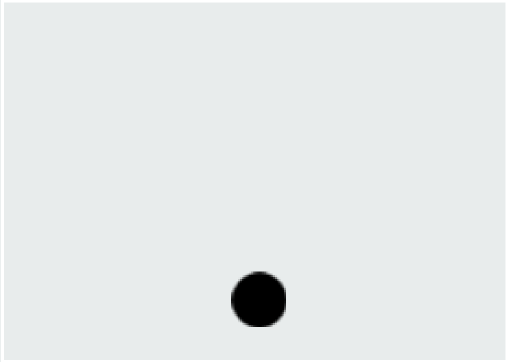
This toolbar contains all the layers that the user can create. Each type of layer will either generate geometry, or apply an effect on the existing model. If you click on one of these buttons the new layer is added at the top of the current stack. Alternatively you can drag one of these buttons to the desired location in the stack.



These controls on the top right corner help re-center and navigate the scrollbar area of the layer compositing panel [the large work area below the toolbar].

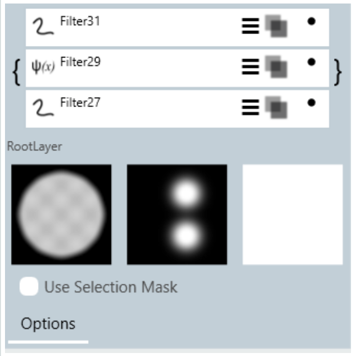


Layers are represented as a stack of tabs [above left]. Each layer has properties and parameters that the user can adjust, and these parameters are accessible through pop-up panels when clicking on the various buttons of each layer's tab.



Drag a layer to the black dot at the bottom of the window to **delete** it.

3.1.1 The Layer Compositing stack

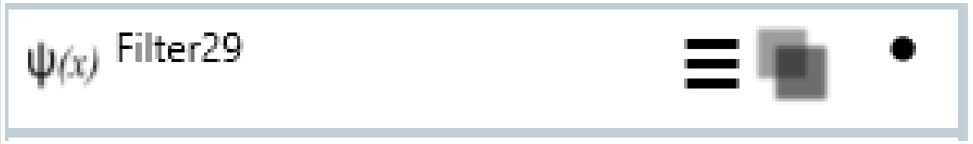


The stack of layer tabs represents the order in which geometries are added together and effects applied from bottom to top. You can drag the tabs in order to rearrange layers. Or drag icons from the toolbar at the top in order to insert new layers.

Below the stack there are two images that display the voxel values at a horizontal slice through the model. The two images represent the shape field and the material ratio gradient respectively. A third image is used for masking operations similar to an alpha channel. The z level of the horizontal slice is determined by the slider at the top of the 3d viewport.

The **options** menu will allow you to **clear** all the layers or **collapse** them into a single voxelized image. This collapsed image contains a snapshot of the current model but it loses completely all the layer information and therefore the ability to edit parametrically the model.

3.1.2 The layer tab



Layers are represented by tabs in the stack panel. Each layer corresponds to a parametric operation that either creates or modifies the model.

The tab gives you access to the layer's properties.

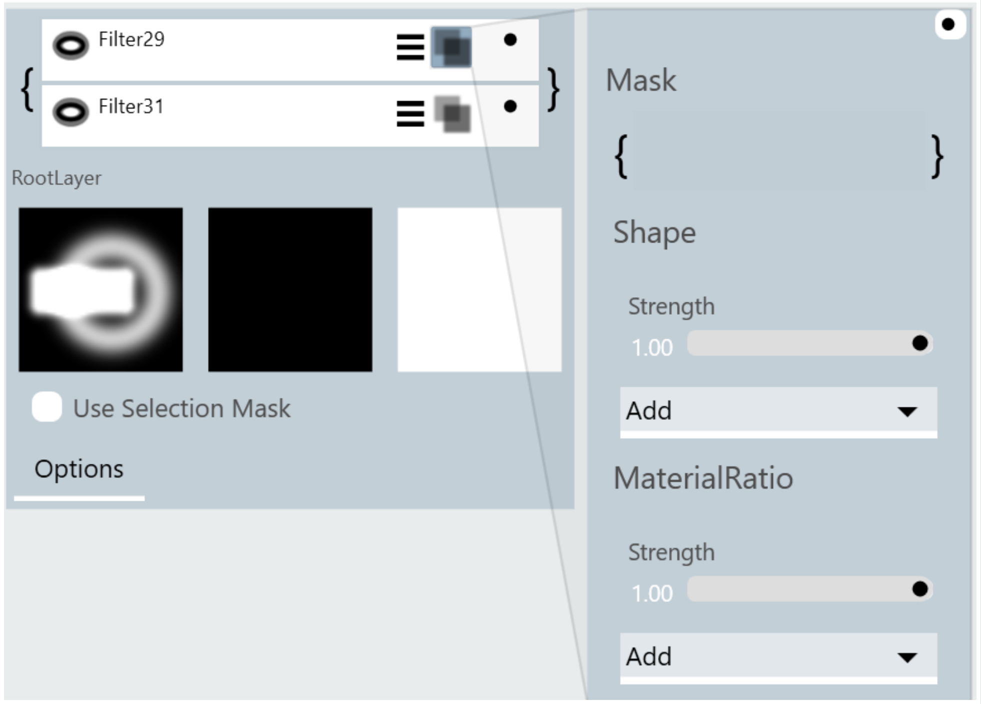
You can **drag the layer from its icon** to change its place in the stack or delete it.

☰ This button will open up the layer's **settings** window as a popup panel on the side. These settings will vary for different types of layers

This button gives access to the layer's **compositing options**, including its blending strength with the layers below.

● The dot can be turned on and off to **enable or disable** the effect of a layer.

3.1.3 The layer compositing options panel

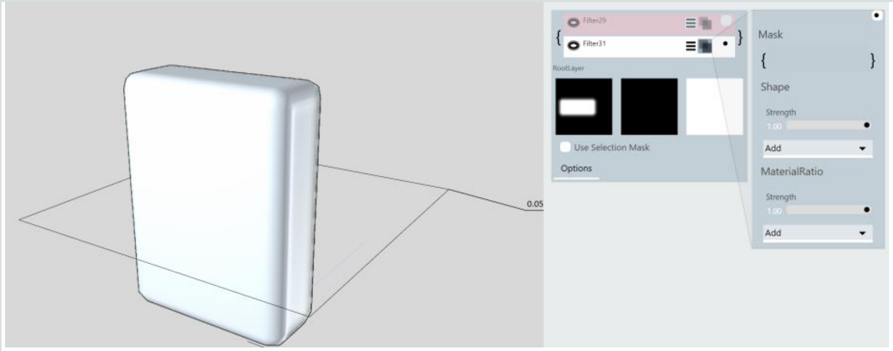

Filter29
Filter31
RootLayer
Use Selection Mask
Options
Mask
{ }
Shape
Strength
1.00
Add
MaterialRatio
Strength
1.00
Add

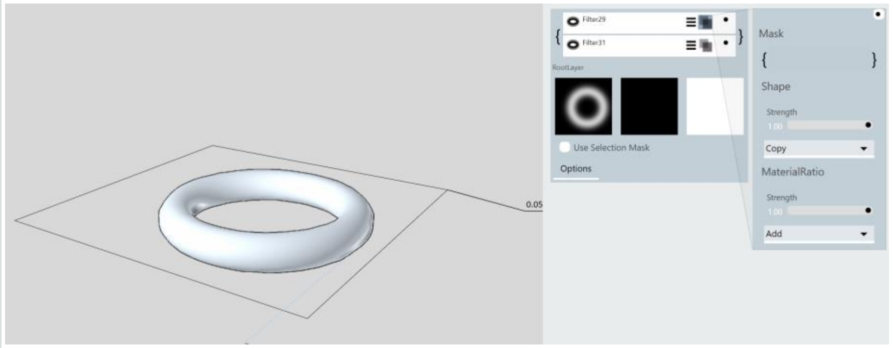
Monolith uses a layer compositing system. This means that every layer generates some voxel values that are subsequently combined with the corresponding voxel values below. The combination can yield different geometric outcomes depending on the arithmetic operation between the layers. Using the layer compositing options pop up you can select the type of compositing operation as well as the strength of composition. The four most commonly used operations are **“copy”, “disabled”, “add”, “subtract” and “multiply”**. The compositing mode can be adjusted separately for each channel.

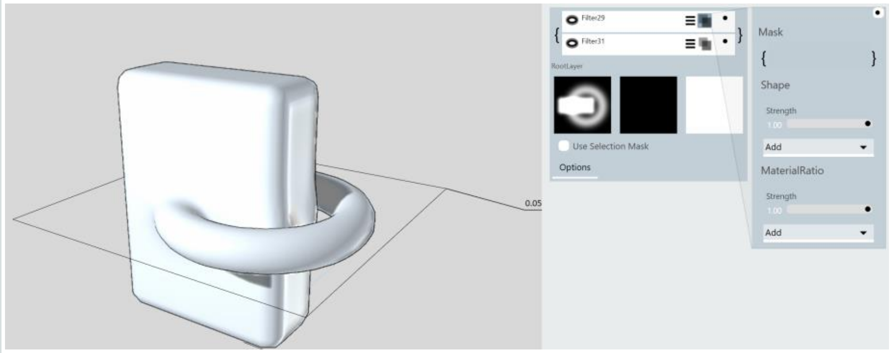
“**Copy**” will replace the layer underneath with the new values. This is the default behaviour for layers that either distort or filter the values of the layer below [e.g. Gaussian blur].

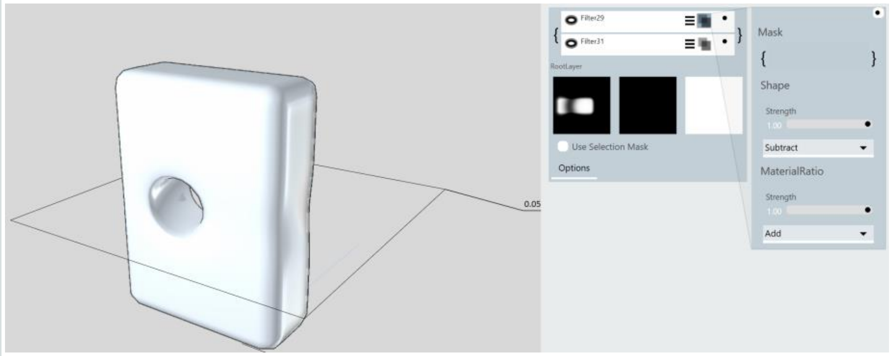
“**Disabled**” will ignore the channel during compositing. This is useful when you want to apply a filter on only a single layer of the underlying image.

“**Add**”, “**Subtract**” and “**Multiply**” will roughly correspond to the geometric Boolean operations of “**union**”, “**difference**” and “**intersection**”

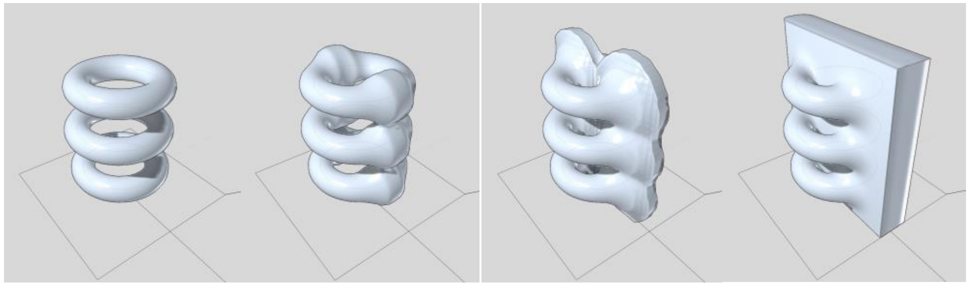

Filter29
Filter31
RootLayer
Use Selection Mask
Options
Mask
Shape
Strength 1.00
Add
MaterialRatio
Strength 1.00
Add


Filter29
Filter31
RootLayer
Use Selection Mask
Options
Mask
Shape
Strength 1.00
Copy
MaterialRatio
Strength 1.00
Add

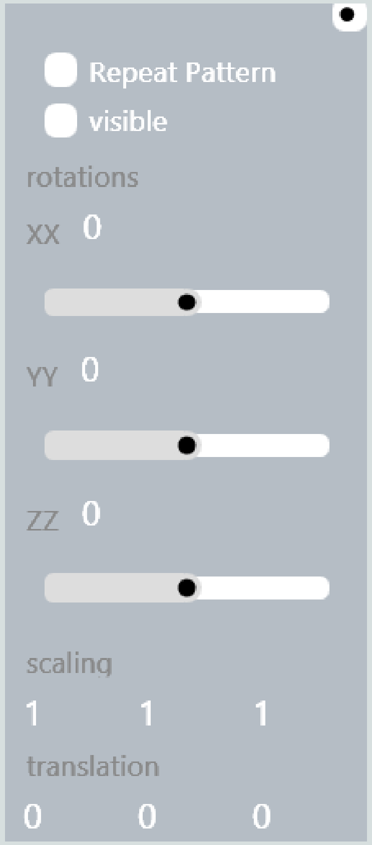

Filter29
Filter31
RootLayer
Use Selection Mask
Options
Mask
Shape
Strength 1.00
Add
MaterialRatio
Strength 1.00
Add


Filter29
Filter31
RootLayer
Use Selection Mask
Options
Mask
Shape
Strength 1.00
Subtract
MaterialRatio
Strength 1.00
Add

For all compositing modes you can also define the strength value to determine the amount of blending:



3.1.4 Layers with transformations



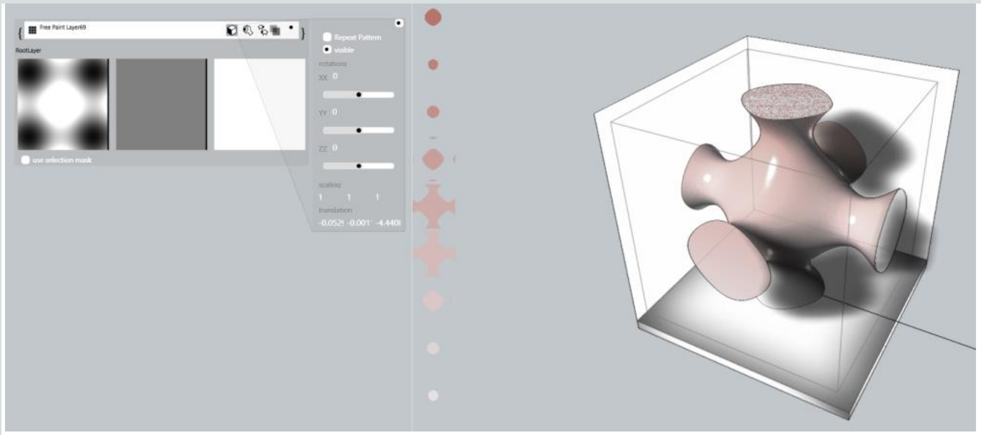
Repeat Pattern
visible
rotations
XX 0
YY 0
ZZ 0
scaling
1 1 1
translation
0 0 0

Some types of layers can be transformed as a whole by applying typical translation, rotation and scaling transformations. In fact any layer that does not have a transformation of its own can be placed within a transformation stack in order to modify its location, scaling and rotation.

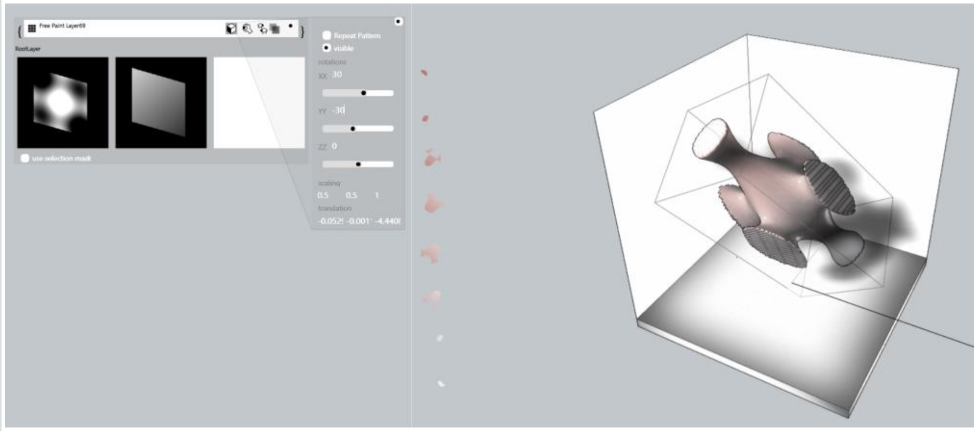
Layers with the affine transform icon can be manipulated in this way. If you click on this icon the transform panel appears [image on the left]:

Use this panel to define the rotation, scaling and translation of the corresponding layer. You can define the translation of transformed layers interactively in the 3d viewport. To do this check the "visible" check box at the top of this panel. This will show you the transformation bounding box in the viewport. You can drag this box around to reposition it.

The "repeat pattern" checkbox converts a layer into a pattern. If you scale down such a layer so that it becomes smaller than the voxel image box in the viewport then this layer will be repeated to fill the complete voxel box.

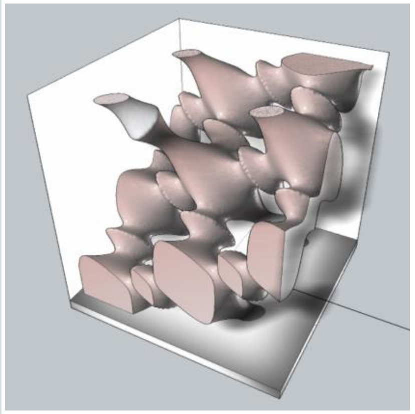


Repeat Pattern
visible
rotations
XX 0
YY 0
ZZ 0
scaling
1 1 1
translation
-0.052! -0.001 -4.440!

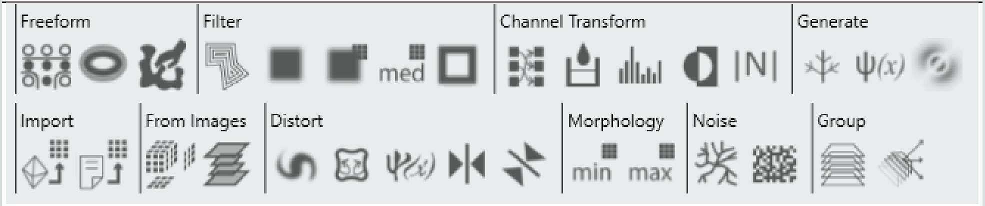


Repeat Pattern
visible
rotations
XX 30
YY -30
ZZ 0
scaling
0.5 0.5 1
translation
-0.052! -0.001 -4.440!

A layer and its transformed version [above]. Same transformed layer with thr "repeat" option turned on [below]:



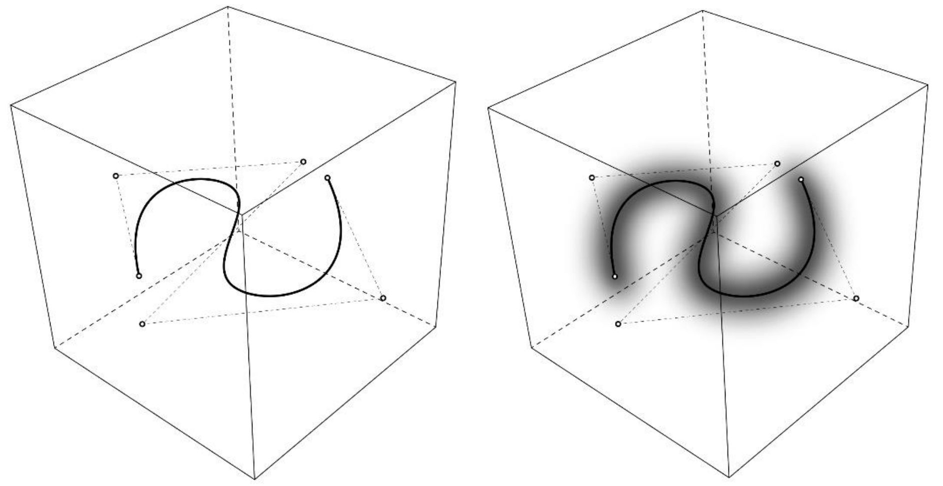
3.1.5 Layer Types overview

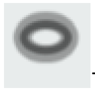


In monolith you can create many types of geometric objects and effects by selecting different tools from the tool bar shown above. Some of these layers are simple effects with minimal user interaction required while others open up extensive user interfaces that enable new workflows and higher customization.

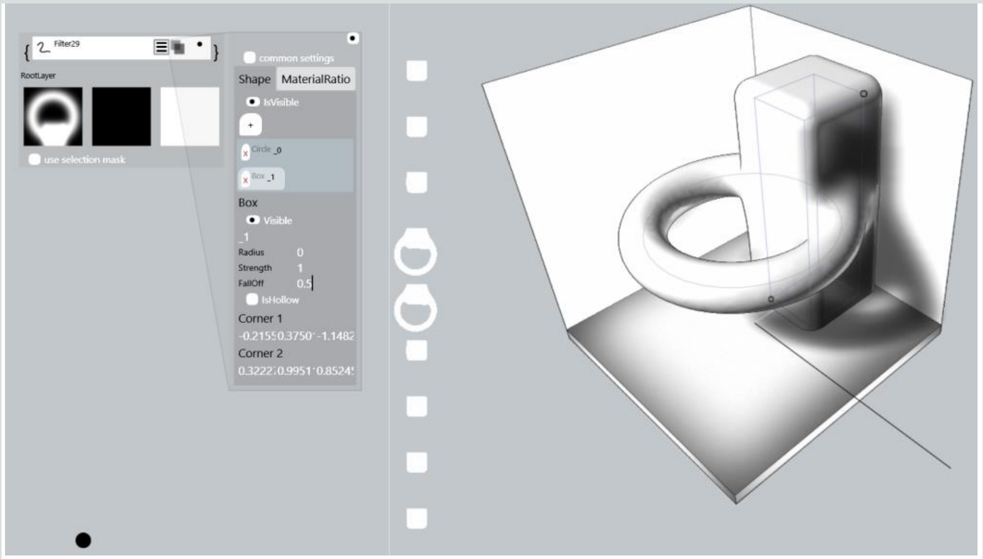
3.1.5.1 Direct modelling layers

3.1.5.1.1 Geometry based modelling

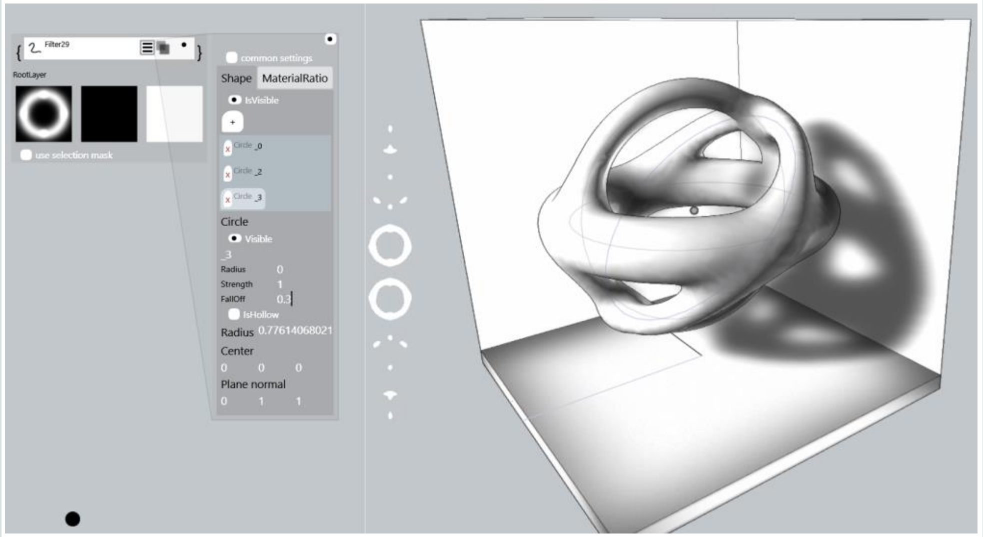



The geometry based layer creates a voxel field controlled by geometric primitives like points, lines, circles, planes, curves and boxes. Each of these primitives generates a halo around it that fades with distance. The strength and falloff of this halo will determine the blending behaviour of the generated field. As objects intersect their fields blend smoothly. Objects with negative strength form voids in the field.

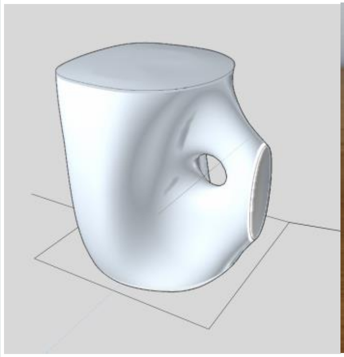
Click on the **+ icon to add new geometric objects** then select them from the list to modify their properties. You can also interact with them directly in the viewport.



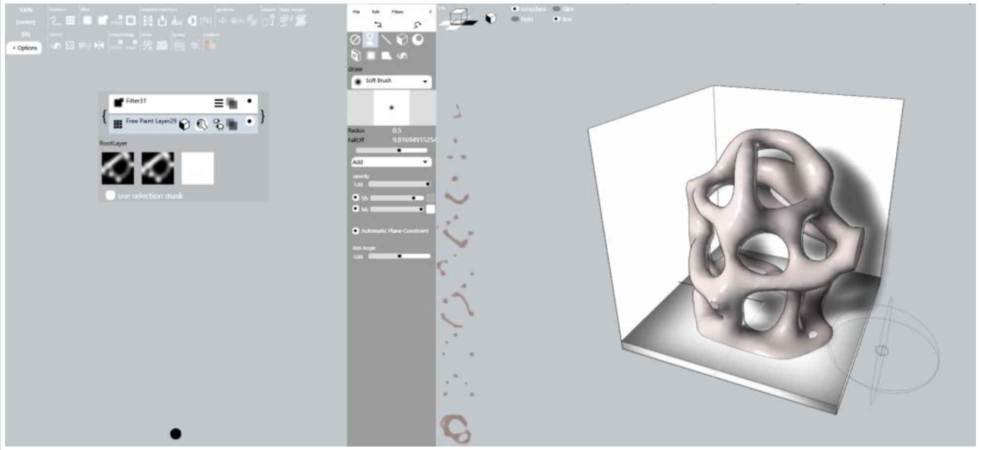
IsVisible
Circle _0
Box _1
Box
Visible
_1
Radius 0
Strength 1
FallOff 0.5
IsHollow
Corner 1
-0.2155 0.3750 -1.1482
Corner 2
0.3222 0.9951 0.8524

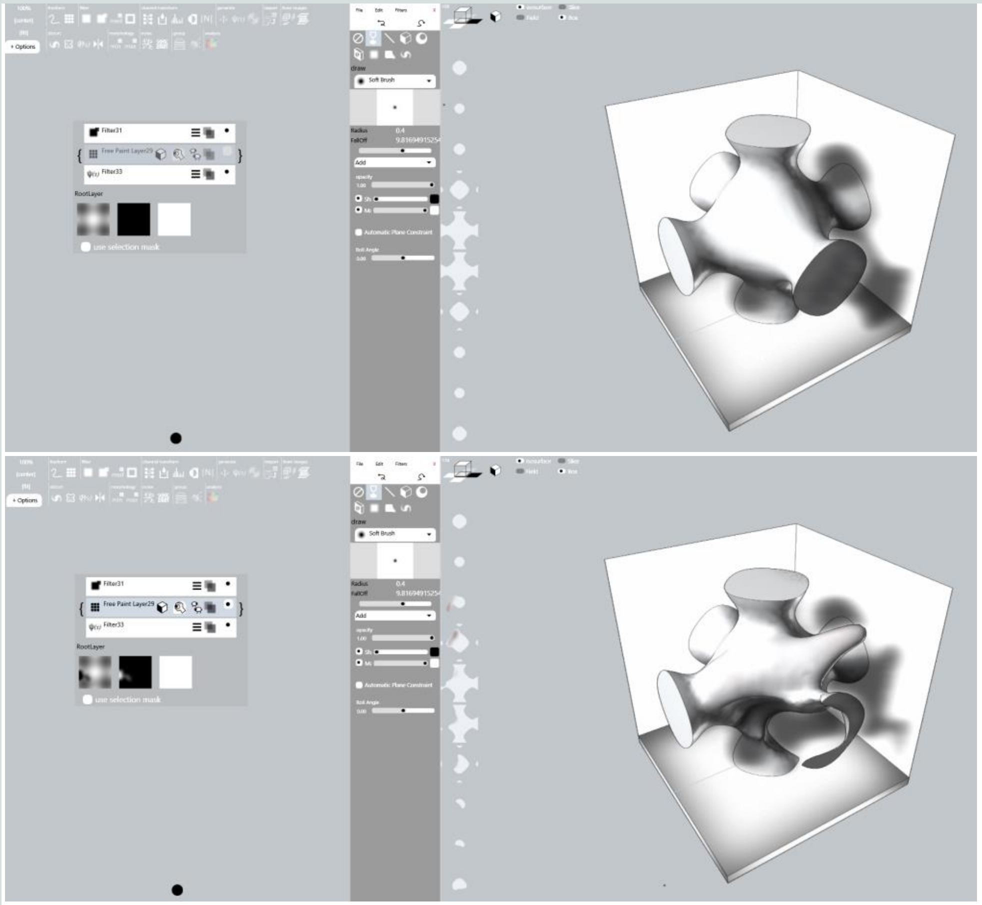


IsVisible
Circle _0
Circle _2
Circle _3
Circle
Visible
_3
Radius 0
Strength 1
FallOff 0.3
IsHollow
Radius 0.77614068021
Center
0 0 0
Plane normal
0 1 1



3.1.5.1.2 Freehand editing layer





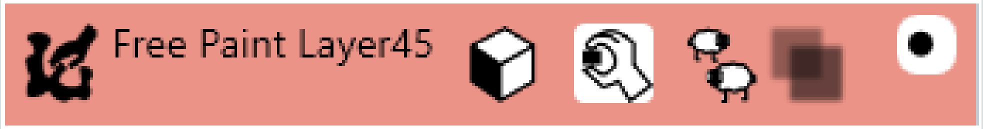
Freehand layers manage their own voxel image and are the closest to the functionality of a conventional image editing application. You can freely paint and create shapes on them by adding or subtracting shape density or altering the material ratio. All filters available to monolith are also applicable on free paint layers.

You can use freehand layers to create geometry form scratch or touch up underlying layers. The default compositing mode for paint layers is “ADD”. This means that if you create a freehand layer on top of an existing layer, while you are painting with a positive strength brush it will seem as if you were adding material to the underlying layer while using a negative strength brush would remove material [without really altering the original layer underneath] [see images on the right where a freehand layer is composited on top of a function based layer].

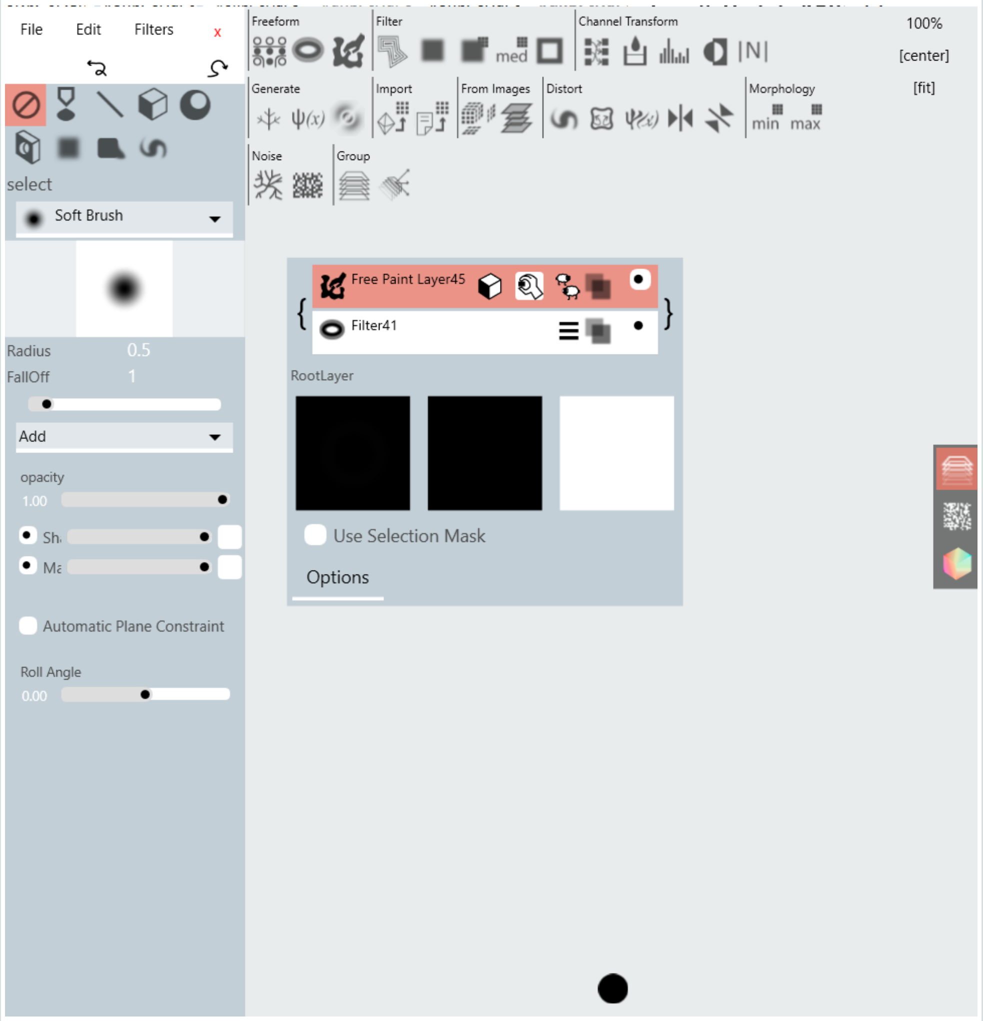
Freehand layers are not parametric, in the sense that all changes you make to them [painting, adding geometry, applying filters] are persistent. They just modify the underlying voxel image without maintaining a history of operations, save from a few undo steps.

3.1.5.1.2.1 Editing freehand layers

In order to start editing a freehand layer click on the edit icon on the layer's tab:

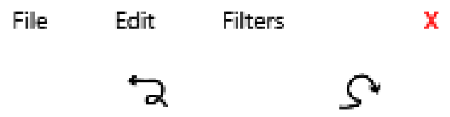


This will bring up the layer editing panel:



The freehand layer editing panel contains all the controls for manipulating the underlying voxel image managed by the selected layer. It is divided into 4 section: the menu, freehand tools, brush and brush alignment.

3.1.5.1.2.1.1 Freehand Menu



The menu bar of the editing panel contains the following functions:

File: Is used to import and export the contents of the freehand layer as a vol file

Edit: here you can change the size and resolution of the underlying voxel image as well as its channel layout. This resolution is independent of the one used for the main application. If the main application has a lower resolution than the freehand layer then you won't be able to see all the details of that layer because it will be downsampled when it gets composited with all the other layers.

Filters: This is the list of all filters available to monolith. They are identical to the filters on the layer selection toolbar but when applied on the freehand layer they modify its voxel values directly rather than create a parametric layer.

: undo and redo.

X : close panel and exit freehand layer editing mode

3.1.5.1.2.1.2 Freehand tools

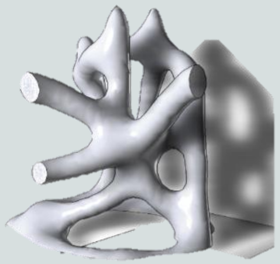
Freehand tools allow you to directly manipulate the voxels in the layer using methods familiar form image editing software and free form sculpting. The effect of most of these tools depends on the currently selected brush and brush properties.

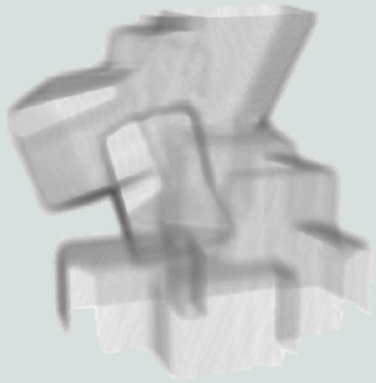
3.1.5.1.2.1.2.1 No tool

The purpose of this button is to unselect any currently selected tool so that normal viewport mouse interaction can resume [for example dragging layers and geometric objects without drawing]

3.1.5.1.2.1.2.2 Paint

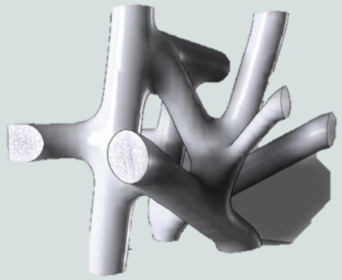
Using this tool you can freely add and remove material from the image. It uses the current brush to determine the profile and fall off of the strokes.





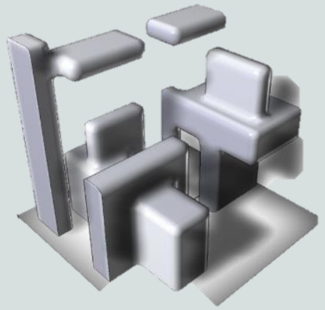
!!! Use the automatic plane constraint to make the brush stroke align with the view. When this switch is ON, if you press the mouse a drawing plane is created that passes through the projected three dimensional position of the mouse and is either parallel to the camera or parallel to the XY plane [if you are looking at the drawing from roughly a top down perspective of more than 45 degrees]. When this switch is OFF the mouse always snaps to the depth of the closest solid surface under the cursor. The effect of having the switch ON is more akin to freely tracing a trajectory in space. Having this off is equivalent to adding or piling up material on the current cursor spot.

3.1.5.1.2.1.2.3 Line



The line tool creates linear brush strokes between two points. [The line drawing is not a click and drag operation, it requires two clicks, on two different locations. This is so that you can rotate and zoom the view in between selecting the end points]

3.1.5.1.2.1.2.4 Box



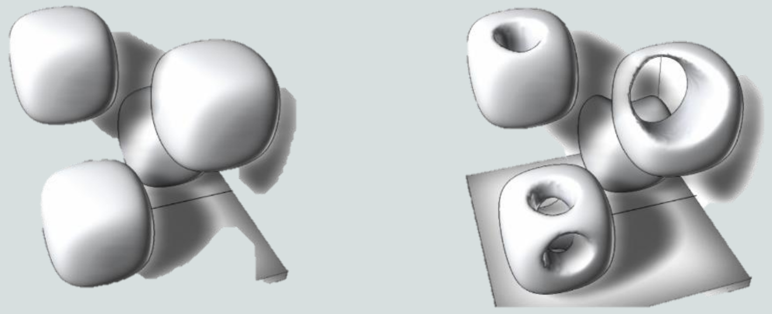
Use this tool to create box shaped solids and voids. The first two clicks will determine the base of the box and the final click the height.

3.1.5.1.2.1.2.5 Sphere

Use this tool to create spherical solids and voids.

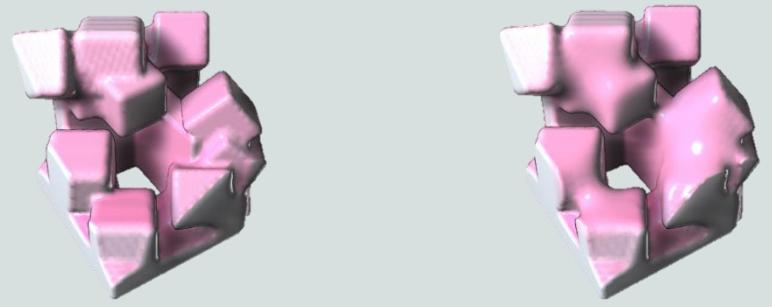
3.1.5.1.2.1.2.6 Drill

Use this tool to drill holes all the way through the voxel image. The tool picks up the tangent plane under the mouse cursor and will drill in the direction normal to that plane.



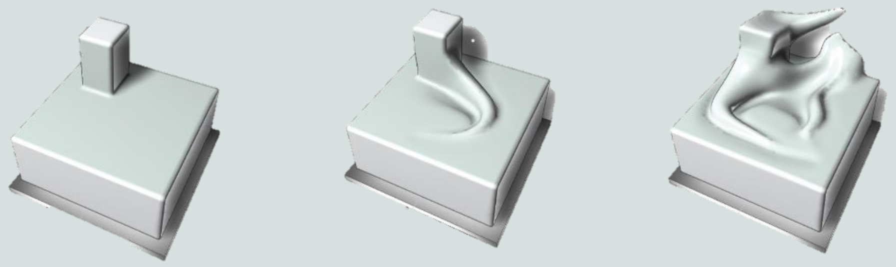
3.1.5.1.2.1.2.7 Smooth

Use this tool to apply local smoothing on the underlying voxel image.



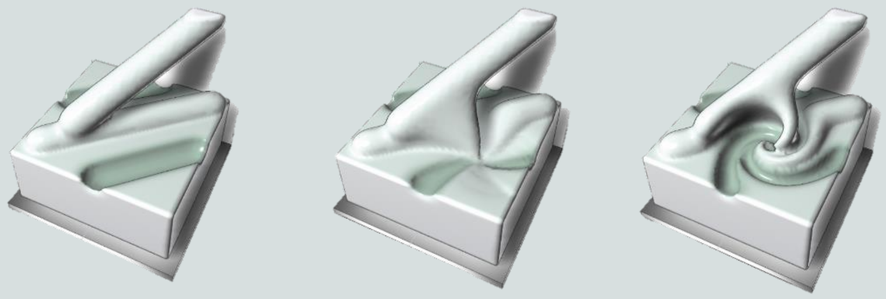
3.1.5.1.2.1.2.8 Smear

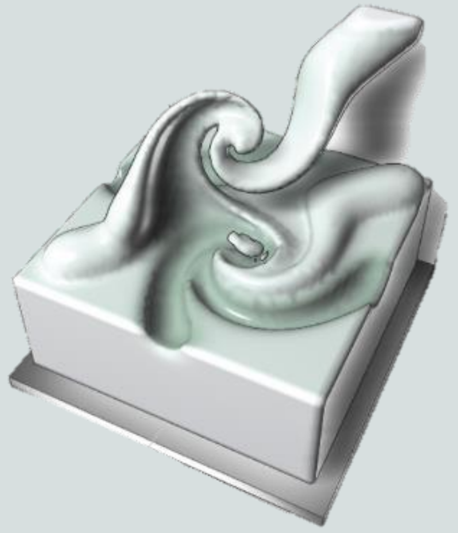
The smear tool drags the material along the mouse path. The effect is similar to clay modelling.



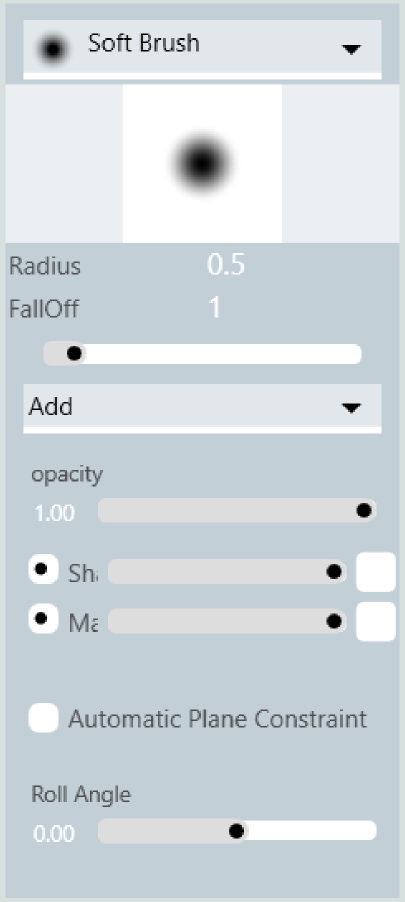
3.1.5.1.2.1.2.9 Twist and pinch

Use this tool to twist, pinch or punch locally the voxel field.





3.1.5.1.2.1.3 Brush settings



Soft Brush

Radius 0.5

FallOff 1

Add

opacity 1.00

Sh

Ma

Automatic Plane Constraint

Roll Angle 0.00

Most of the freehand editing tools rely on a brush to determine the shape and blending of the area of influence around the mouse cursor. You can select a brush type from the combo-box below the tools.

As the mouse cursor moves the brush is automatically aligned with the underlying geometry. This is easier to see with the box brush as it will always paint protrusions perpendicular to whatever solid surface is under the mouse. The roll [in plane angle] of the brush is determined by the “Roll angle” slider. The default roll is to align the angle of the brush with the viewer so that the local Y of the brush is aligned with the viewport’s vertical direction.

The combo box below determines the compositing mode for paint operations. The opacity will determine the strength of the effect.

Below there will be a number of slider equal to the number of channels in the voxel image. Typically there are just two sliders for the shape and material ratio channels. These determine the “colour” values of the brush as it paints over the corresponding channels. Unlike 2d painting programs here you can set negative values to the brush colour which would correspond to subtraction of material.

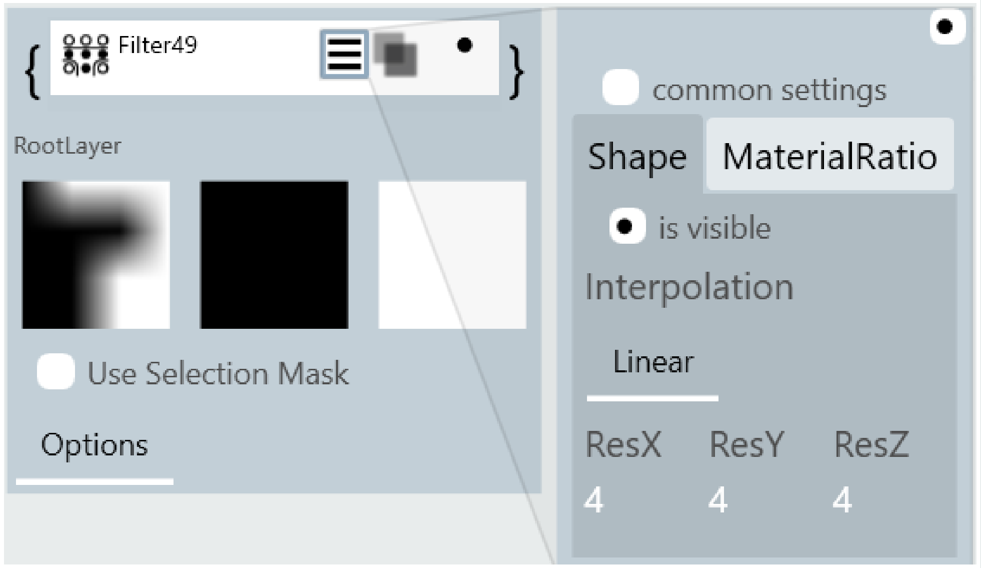
To the right of each slider there is a switch [the black dot] you can click there to deactivate painting on a channel. For example you may want to paint the material ratio field without modifying the shape of the object.

Finally below the channel sliders are the brush orientation settings. The automatic plane constrain switch, determines how click and drag operations work with the brush. If this switch is off the brush will just paint on the volume under the mouse cursor thus piling material over existing material.

If the automatic plane constraint is on, as soon as you press the mouse a temporary drawing plane is constructed that passes through the mouse cursor and is either aligned to the camera or to the xy plane if the camera has a more than 45 degrees angle from the xy plane [looking at the scene from above or below]. As you drag the mouse the cursor intersects this temporary plane in mid-air without the need of some underlying geometry to support it.

3.1.5.1.3 Grid point editing layer

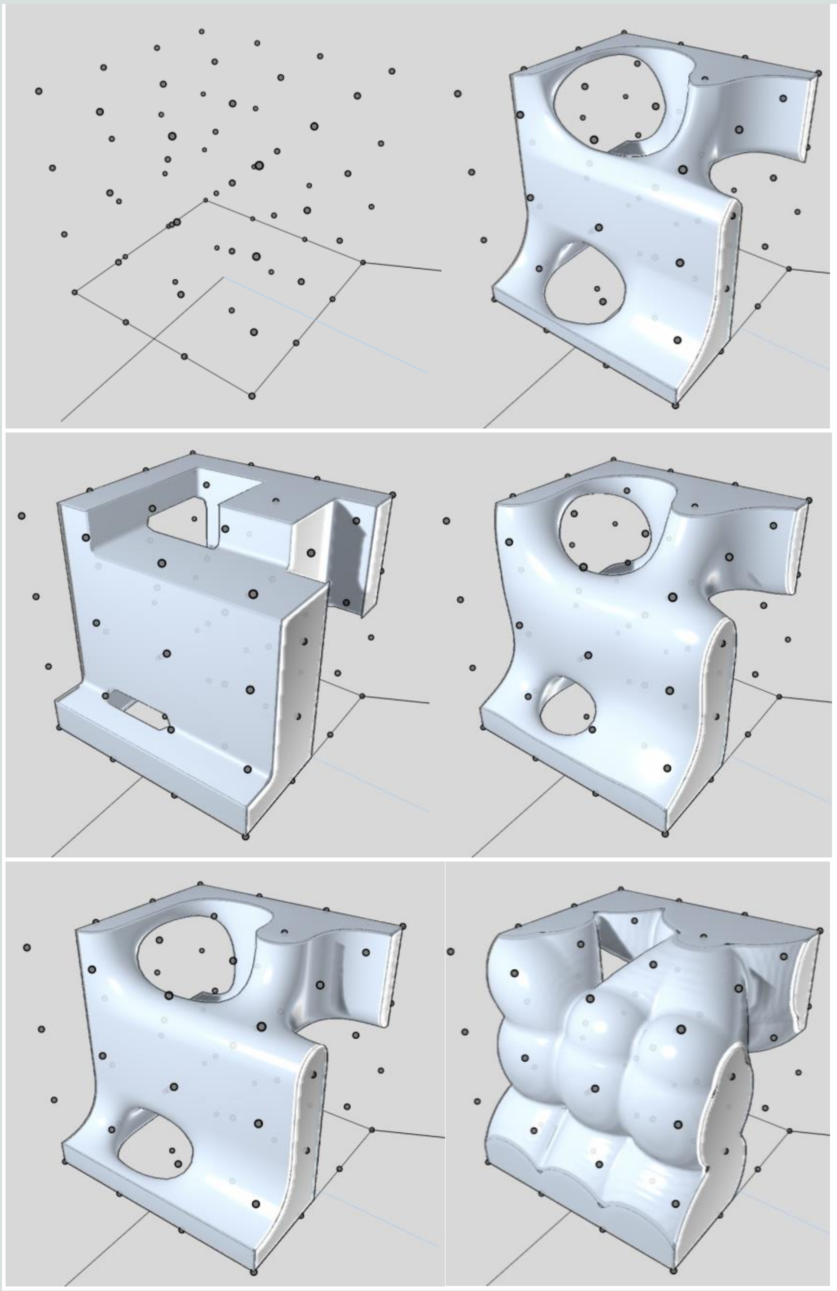
In grid editing mode you interact with a three dimensional grid of points. By determining a weight factor for each point, you determine whether the point is inside or outside the solid and how it would blend with its neighbours. You can right click on points to switch them from completely solid to completely void or the other way around. By left clicking and dragging up or down you can have a more granular control over the strength of each point.



The grid editing control panel itself has only a few options. Most important is the **visibility switch** for the grid points at the top.

In addition you can choose and **interpolation mode** between points which will determine the outcome of the point to point blending. On the right of this page you cans see some examples of the effect of different interpolation modes on the final shape.

You can also change the **grid resolution** here if you need more detail. It is a good idea to start with a low resolution grid to set out the large scale shapes and then increase the resolution if you want to add more detail.

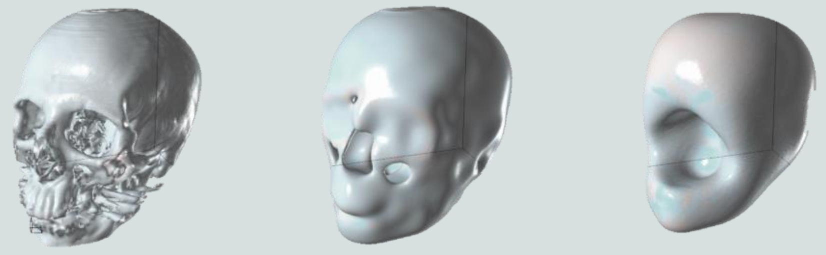


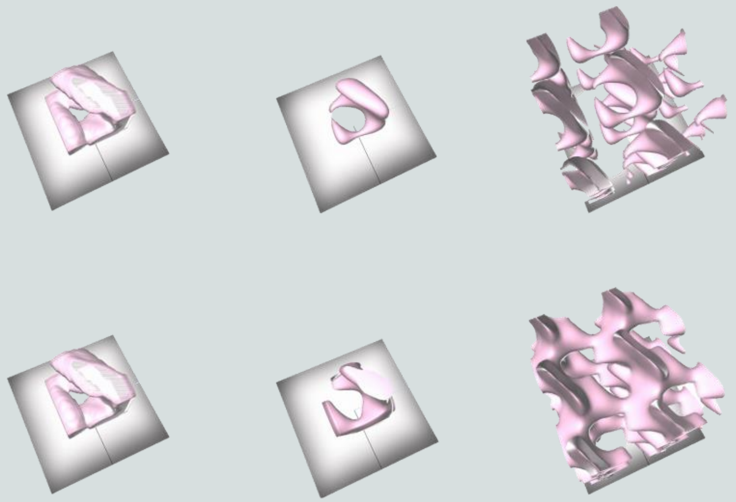
3.1.5.2 Filters

Most layers types do not generate new geometry but rather affect the layers below their position within the layer stack. Filter based layers extend the common 2d image filters to 3 dimensional voxel fields.

3.1.5.2.1 Gaussian blur

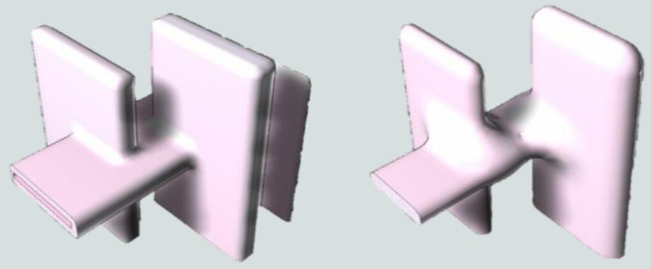
Gaussian blur is a smoothing technique that can remove noise, texture or small features from a voxel image. You can also apply a periodic Gaussian blur thus ensuring a seamless tiling of the blurred image [see pink example above].

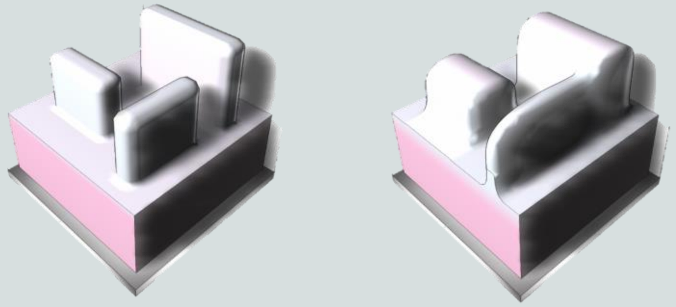




3.1.5.2.2 Kernel based blur

Unlike the Gaussian blur filter, the kernel based version can use a variety of smoothing kernels including box, sphere and Gaussian. It is in general slower than the Gaussian filter for large kernels but it enables anisotropic blurring which is not supported for the Gaussian filter [see second example above].



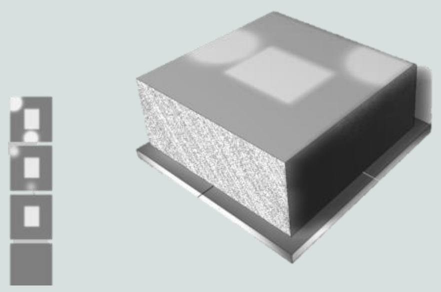


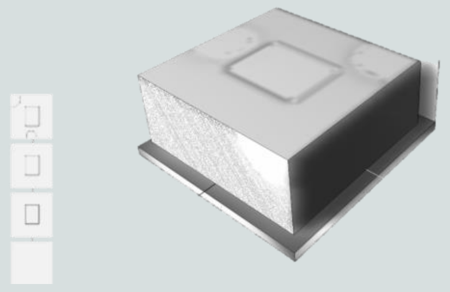
3.1.5.2.3 Median

The median filter replaces each voxel value with the median of its neighbours. The effect on the shape channel is similar to smoothing. It is more useful in removing artefacts from the material ratio channel as it can reduce noise while preserving sharp edges.

3.1.5.2.4 Laplacian

The laplacian filter is basically a three dimensional edge detection. Regions of the voxel image that display rapid change [sharp edges] will become highlighted by the laplacian filter.



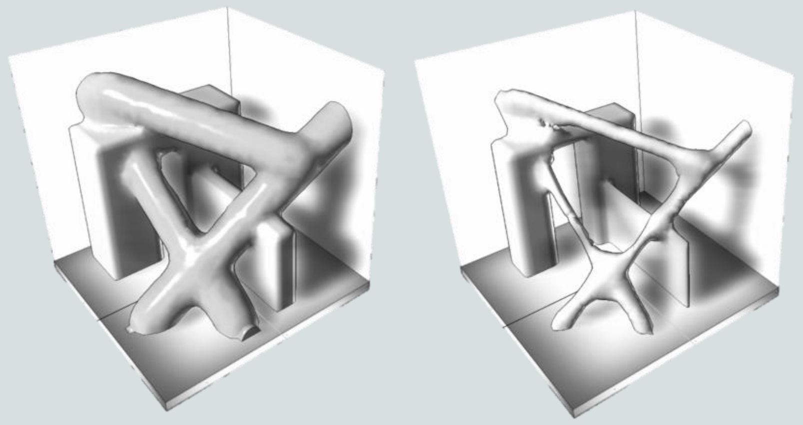


3.1.5.3 Morphological operators

These are the two classic image morphological operators that can be combined to achieve various typical image processing effects [erosion, opening, closing etc…]

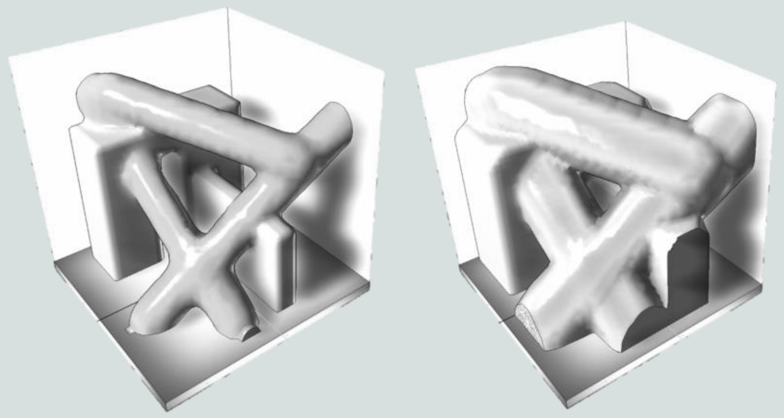
3.1.5.3.1 Minimum

The min filter replaces the value of each voxel with the smallest value in its neighbourhood. The result resembles erosion or skeletonization of the shape.



3.1.5.3.2 Maximum

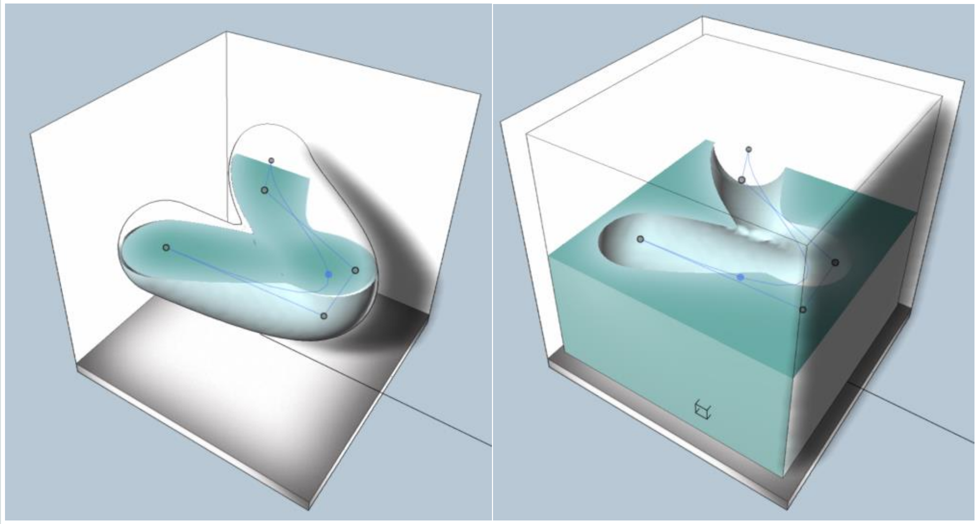
The max filter replaces the value of each voxel with the largest value in its neighbourhood. The result resembles a bloating or inflation of the shape.



3.1.5.4 Channel transformations

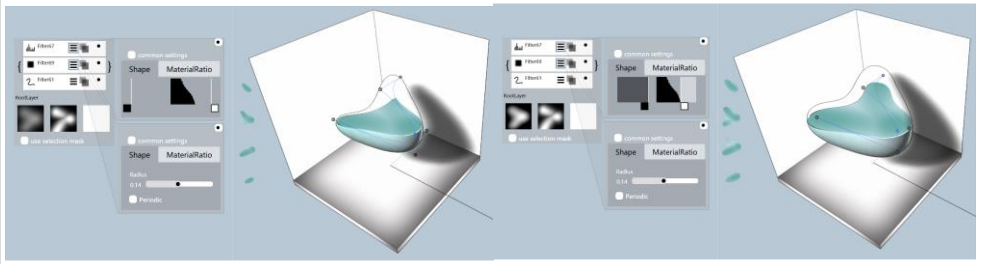
3.1.5.4.1 Invert

This filter replaces each value in the image with (1-value) therefore swapping solid and void regions [provided the image is normalized in the 0 to 1 range first]



3.1.5.4.2 Histogram

The histogram filter is used in order to analyse the distribution of densities in the voxel field. In addition it allows for the renormalization of the field within a selected range.



3.1.5.4.3 Channel remap

Use channel remapping to shuffle the channels of the underlying layer. For example you may want to swap the shape field with the material ratio or the mask field.

3.1.5.4.4 Fill

The fill filter simply fills the channel with a solid value. Useful also when clearing the free paint layer.

3.1.5.4.5 Normalize

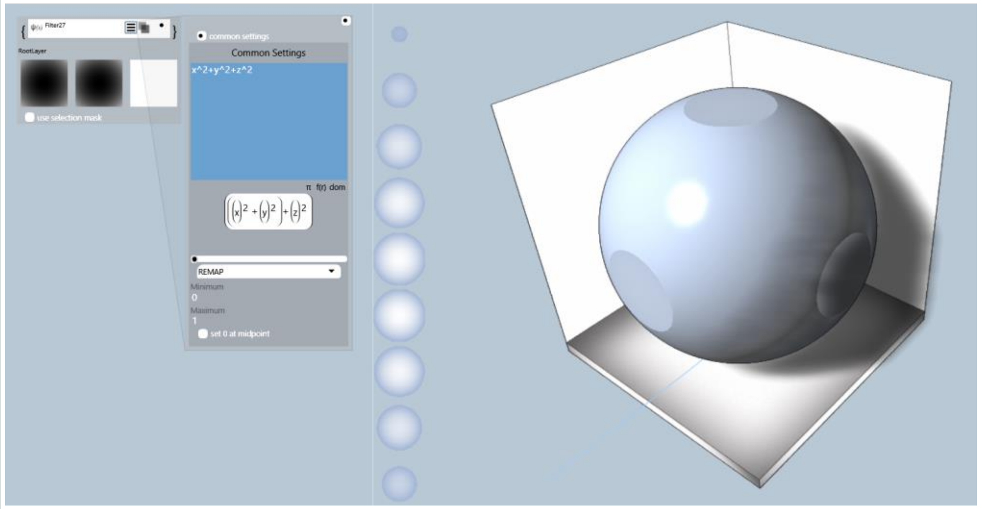
The normalization filter will bring the voxel image values within the range of 0 to 1 or -1 to 1. This is useful in cases where you generated an image with some process that results in very high field values or a large range [e.g. visualization of stresses in a solid] or after you apply a Gaussian blur which in general reduces the intensity of the image.

3.1.5.5 Generative

3.1.5.5.1 Function based fields

The function based voxel field generation layer, creates a field whose values are defined by a formula. You can type any formula in the blue text box and it gets automatically parsed and displayed. A few predefined formulas are already provided by clicking on the **f(r)** button.

The purpose of the formula is to generate a value for each voxel in space. For example the formula $x^2+ y^2+ z^2$ will generate a field that has the value 0 in the centre and increases radially with the square of the distance. The isosurfaces of this field are spheres as the equation:
$\boldsymbol{x^2+ y^2+ z^2=const}$ describes a spherical surface.



Filter27

RootLayer

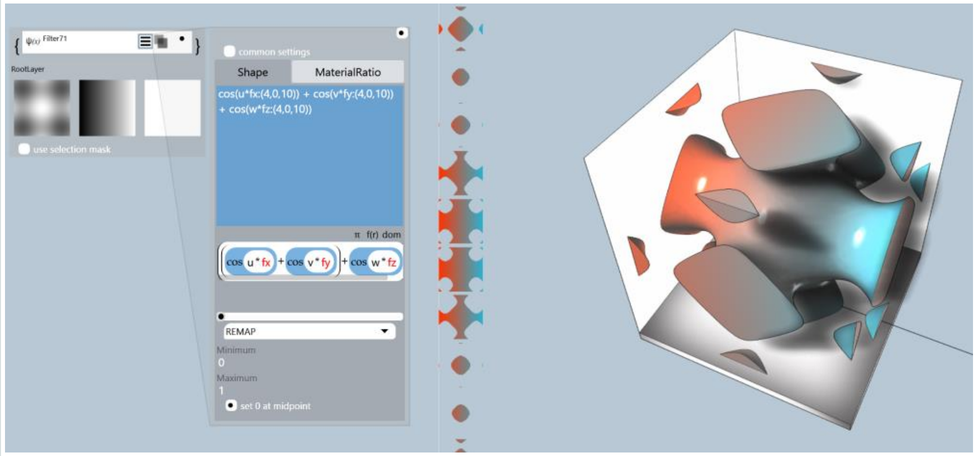
use selection mask

common settings

Common Settings

```
x^2+y^2+z^2
```

Minimum
0
Maximum
1
set 0 at midpoint



Filter71

RootLayer

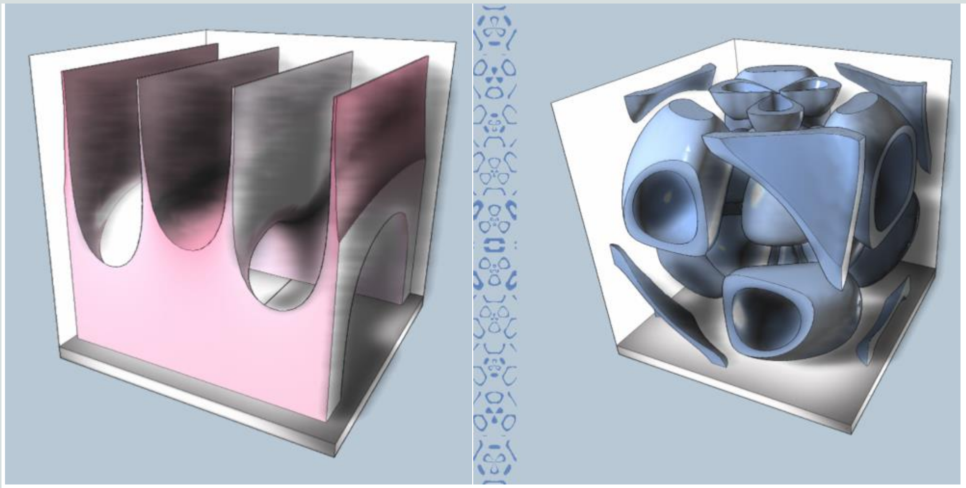
use selection mask

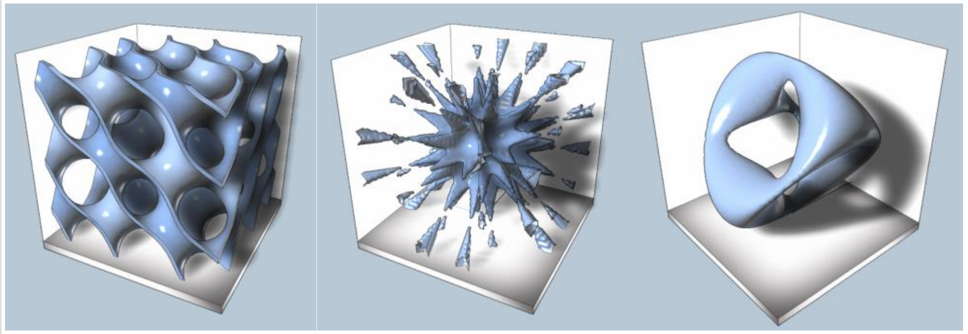
common settings

Shape | MaterialRatio

```
cos(u*fx:(4,0,10)) + cos(v*fy:(4,0,10))
+ cos(w*fz:(4,0,10))
```

Minimum
0
Maximum
1
set 0 at midpoint





When you type a formula, the formula evaluator will step through each voxel in the field, feed the XYZ, UVW, ρϕθ coordinates [absolute, normalized and relative spherical] to the formula, compute the resulting number and assign it to the corresponding voxel’s value.

There are a few variable, function and operator names that are predefined but you can also define your own variables within a formula.

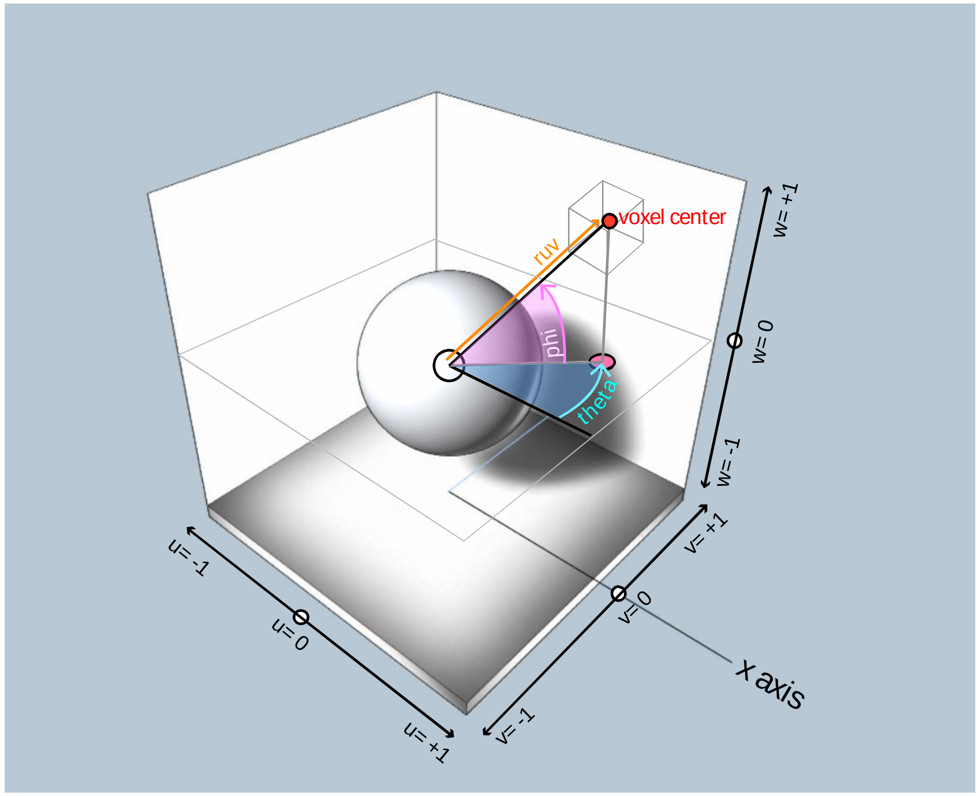
The parser recognizes the following symbols:

3.1.5.5.1.1.1 Constants: e pi

The predefined constants e [for the Euler number] and pi [for π]

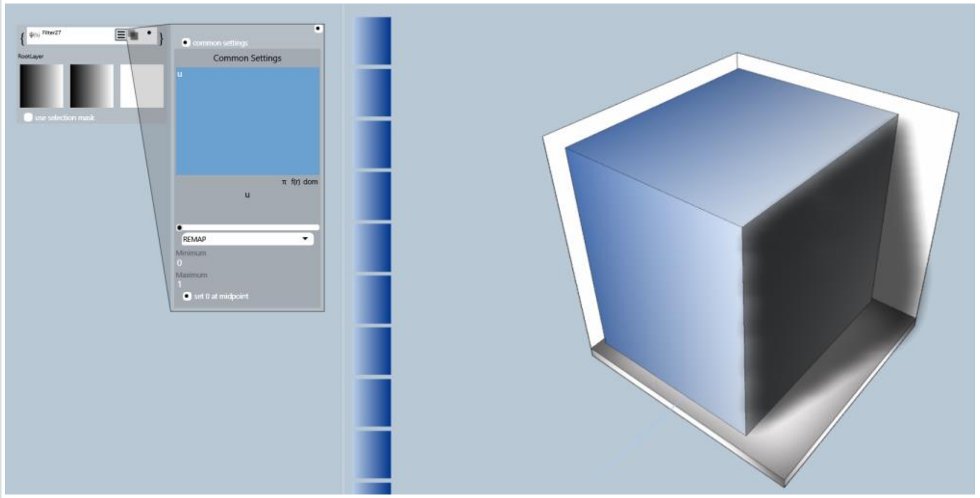
3.1.5.5.1.1.2 Space coordinates: x y z u v w ruv phi theta

For each voxel the evaluator will compute its coordinates in three different systems and pass them through the formula. Each voxel has a location **x,y,z** in inches. These are determined by the size of your printing region. In addition the evaluator computes normalized volume coordinates for each voxel. These are represented by the letters **u,v,w** and vary from -1.0 to +1.0 along each of the x,y,z axes. So the center pixel of the voxel image always has coordinates [0,0,0] and the bottom left corner [-1, -1, -1] etc.. Finally the **ruv, phi** and **theta** represent the spherical coordinates in relation to the UVW system. ruv is the radius [$\sqrt{(u^2+ v^2+ w^2)}$] theta is the angle in the UV plane and phi the angle out of the UV plane.



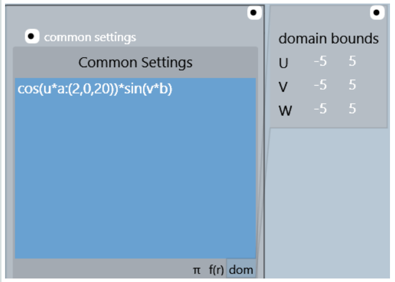
The diagram above shows the uvw and ρθϕ coordinate systems.

Note that the uvw system does not respect the aspect ratio of the box. Even for an elongated non cubic box the uvw coordinates will always span the -1 to +1 intervals. However the xyz system will reflect the actual dimensions of the box. You can freely combine the uvw and xyz variables in your formulas.



In the image above the formula “u” describes a field that varies from -1 to 1 along the x axis which is simply a linear gradient.

You can set the ranges of uvw to something different than [-1,1] by clicking on the **dom** button:



cos(u*a:(2,0,20))*sin(v*b)

U	-5	5
V	-5	5
W	-5	5

!!!Note that certain preset formulas [e.g. gyroid etc..] will set the uvw range to non-normalized intervals when activated.

3.1.5.5.1.1.3 Layer color: d

The predefined variable **d** represents the value of the underlying layer on which the function based filter is superimposed. This can be used in order to make function filters that respond to the underlying geometry.

3.1.5.5.1.1.4 Operators: + - * / ^ %

The typical arithmetic operators for addition, subtraction, multiplication and division.

+	: add
-	: subtract
*	: multiply
/	: divide

*Use parentheses to clarify the order of operations: (u + v)*w   instead of   u + v*w*

^	: power , e.g. x^3 is x^3
%	: remainder of division

3.1.5.5.1.1.5 Basic functions

All the usual basic functions are supported, including trigonometric, inverse trigonometric, hyperbolic and exponential. Arguments for the trigonometric functions are in radians.

sqrt	: square root : sqrt(a)
pow	: raise to power (same as using ^): pow(x, 3.5)
abs	: absolute value : abs(u)

sin	: sine
cos	: cosine
tan	: tangent

asin	: arc sine
acos	: arc cosine
atan	: arc tangent
atan2	: arc tangent of ratio : e.g atan2(y,x) is the same as atan(y/x) but its sine reflects the quadrant. It is better to use atan2 when looking for results in the full circle [-π, +π]

sinh	: hyperbolic sine
cosh	: hyperbolic cosine
tanh	: hyperbolic tangent

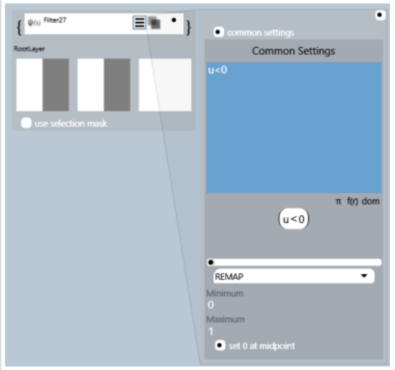
floor	: floor of number : floor(2.6) is 2
ceil	: ceiling of number: ceil(2.6) is 3
round	: round number: round(2.6) is 3 but round (2.3) is 2

rnd	: random number: rnd or rnd() is a random number between 0 and 1, rnd(5) is a random number between 0 and 5

log	: natural logarithm
exp	: exponential : exp(x) is e^x

3.1.5.5.1.1.6 Comparison and logical operators: <> & | !

In the formula editor Boolean results are mapped to the numbers 0 [false] and 1 [true].



u<0

REMAP

Minimum
0

Maximum
1

For example in the formula above the field takes the value 1.0 when u is greater than 0.0 and the value 0.0 otherwise. The result is a step function that jumps from 0 to 1 along the x axis. [u is the parameter that varies with the x axis]

>	: greater than
<	: less than

&	: logical AND
|	: logical OR
!	: logical NOT
xor	: logical XOR [e.g. xor(u>0.5, v<0.2)]

3.1.5.5.1.1.7 Conditional: if

If is a function of three variables. If the first variable evaluates to 1 [true] then if evaluates to the 2nd variable else it evaluates to the 3rd.

***if**(u>0.5, cos(u*3.0), v)*

The above statement says that if the u parameter of a voxel is greater than 0.5 then the voxel value should be the number cos(3 u) else its value should be equal to the v parameter.

3.1.5.5.1.1.8 List operators: norm norm2 sum prod min max

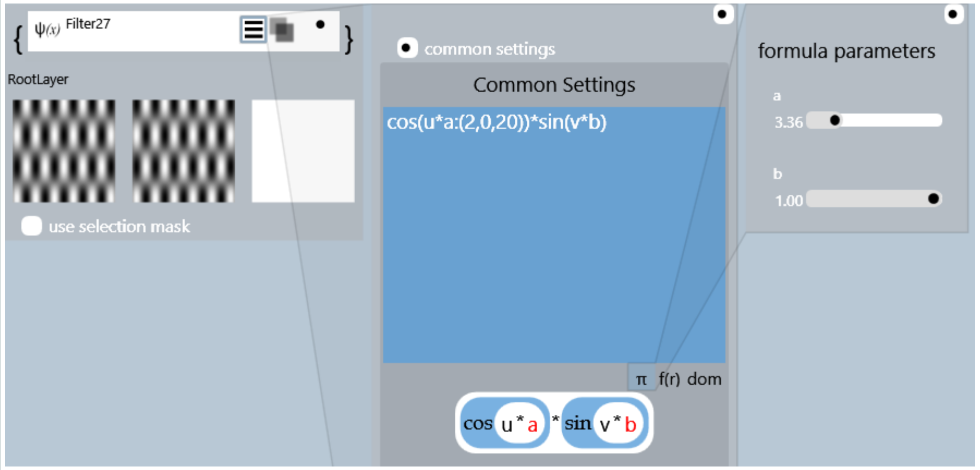
The list operators follow the syntax op(v0, v1, v2 …) that is you can place an arbitrary number of comma separated expressions inside the parentheses following the operator token. These operators are the following:

norm	: return the norm of the list (e.g. norm(x, y) is equal to sqrt(x^2+y^2))
norm2	: is the square of the norm (e.g. norm2(x,y,u,a) is equal to (x^2+ y^2+ u^2+ a^2))
sum	: return the sum of the values in the list (e.g. sum(a,2,b) is equal to a+2+b)
prod	: is the product of the list (e.g. prod(x,y,w,cos(z)) is equal to x*y*w*cos(z))

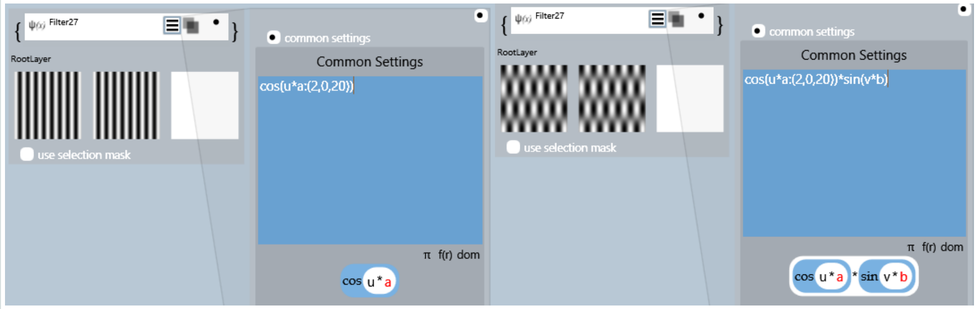
min	: returns the minimum value in the list
max	: returns the maximum value in the list

3.1.5.5.1.1.9 custom variables

The formula evaluator will automatically interpret any unrecognized symbol as a user defined variable. This makes it easy to define parametric formulas and explore their variations. A user variable will appear in red on the formula field under the blue text box and. You can click and drag a red variable to change its value and observe the shape changing or click on the π button to open up the user variable slider panel.



cos(u*a:(2,0,20))*sin(v*b)



cos(u*a:(2,0,20))

cos(u*a:(2,0,20))*sin(v*b)

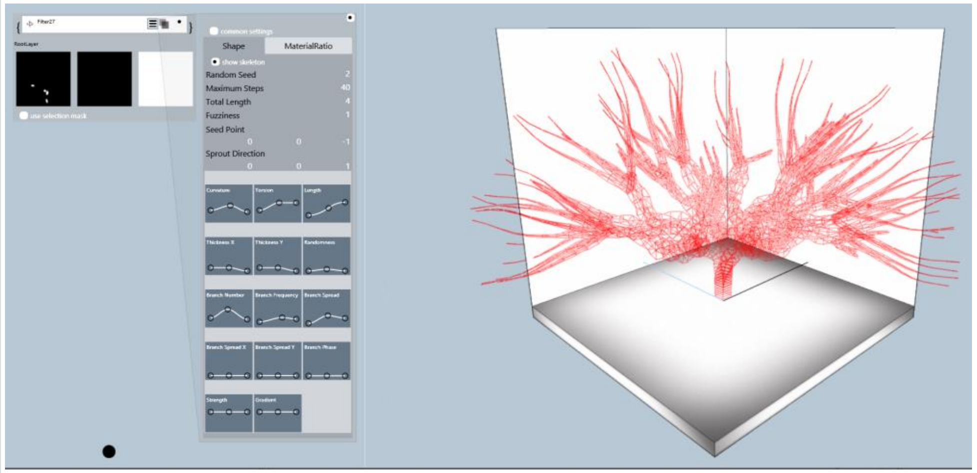
You can also set the default value and range of a custom variable using the **:** operator. For example :

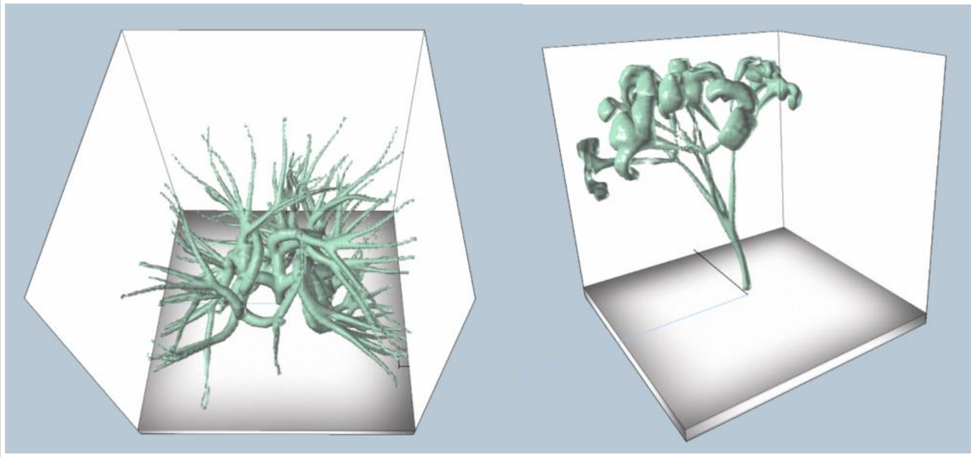
*cos(u*a:(2,0,20))*

is the same as cos(u*a) but a is understood to have the default value 2 and as you drag its slider it will vary from a minimum of 0 to a maximum of 20.

3.1.5.5.2 Branching

The branching filter generates three dimensional tree-like branching structures. The branching pattern and geometric variation in the cross sections of the brunches are determined by special controls that act as mini timelines.



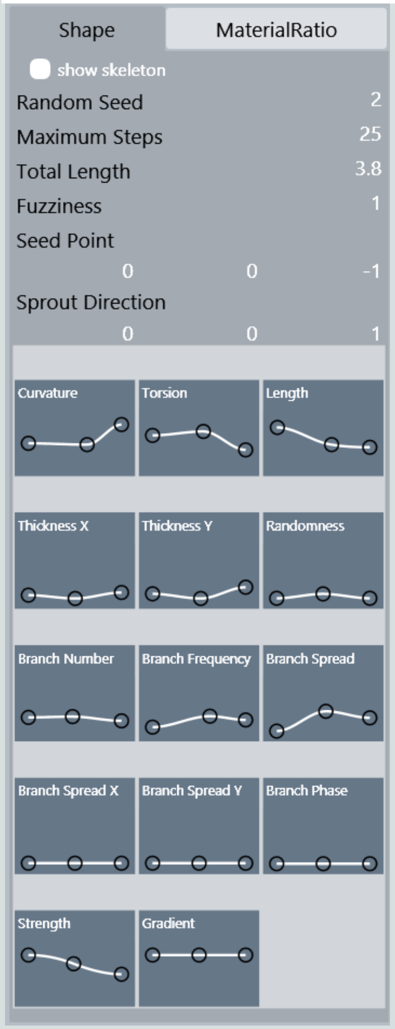


Each of these controls defines a curve which determines how a particular parameter [e.g. width of brunch section or curvature of brunch] varies from the root to the tips of the terminal branches.

When working with complex trees it is a good idea to turn off automatic updating [from the main panel] as branching structures slow down the recalculation of the voxel image significantly.

The control panel for the branching structure is separated into two sections. The top sections determines overall characteristics of the generated pattern while the bottom is a set of micro-timelines that determine the variation of branching properties with respect to growth time.

The following options are exposed in the control panel:



Shape | MaterialRatio

show skeleton

Random Seed		2
Maximum Steps		25
Total Length		3.8
Fuzziness		1
Seed Point		
0	0	-1
Sprout Direction		
0	0	1

Show skeleton: set this to on to preview the skeletal structure of the tree. This is useful because it will allow you to preview the form in real time without having to recompute the voxel image field and the isosurface.

Random seed: any integer number, it is the seed for random number generator used by the random properties of the tree. Two trees with exact same settings and the same random seed will look identically "random".

Maximum steps: the number of growth steps [and generated branch segments] from the root to the tip of the tree. The larger this number the slower the calculation speed.

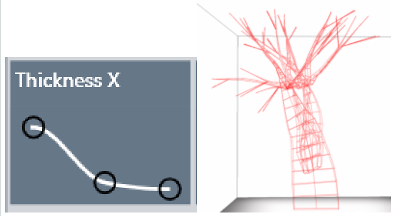
Total length: This is the total length of all the segments from the root to a tip of the tree. This settings defines the maximum height of the tree if it didn't bend at all and therefore directly controls the overall size of the resulting structure.

Fuzziness: this number determines the softness of the field around the tree during voxelization

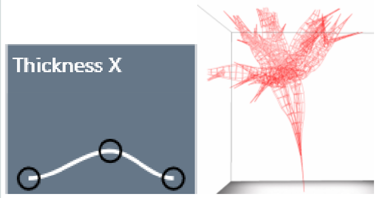
Seed point: these three numbers are the XYZ coordinates of the root node of the tree

Sprout direction: These are the XYZ components of the sprouting vector. The default is [0,0,1] which means that the tree leaves the seed point with a direction along the positive Z axis.

The next section of the control panel is using the micro-timelines to determine the growth pattern of the tree structure. For example in the following image we see the thickness control defining a cross section for the branches that starts quite wide and finishes at very thin tips.



We could edit this timeline to one that starts thin becomes thicker in the middle and collapses again to zero at the tips:



The timelines control the following parameters:

Curvature: Determines how fast the branches bend as they propagate. Higher values will result in more crooked trees with branches forming loops. Also not that it takes both positive and negative values so the neutral line [straight branches] is somewhere in the middle of the graph [pink line].

Torsion: Determines the twist of the branches around their axis of growth.

Length: Determines the length of individual branch segments. The maximum length of a branch segment is equal to totalLength/Steps. By changing this property you can change the rate of growth at particular sections of the structures.

Thickness X and Y: These two timelines control the dimensions of the cross section in fractions of the overall voxel image size.

Randomness: Determines the jitteriness of the growth pattern.

Branch number: Determines the number of branches at branching points.

Branch Frequency: Determines the frequency of branching points. *!!! Be careful with this control, if you set it too low [control points near the bottom of the graph] then it means that the tree will branch out at every single step resulting in an exponential proliferation of branches and possibly a memory overload*. Moving the control points to the upper limit of the graph will make branching less likely.

Branch Spread: The spread angle away from the core branch for new branches.

Branch Spread X and Y: these controls work in relation to the overall spread angle. They determine the spreading anisotropy.

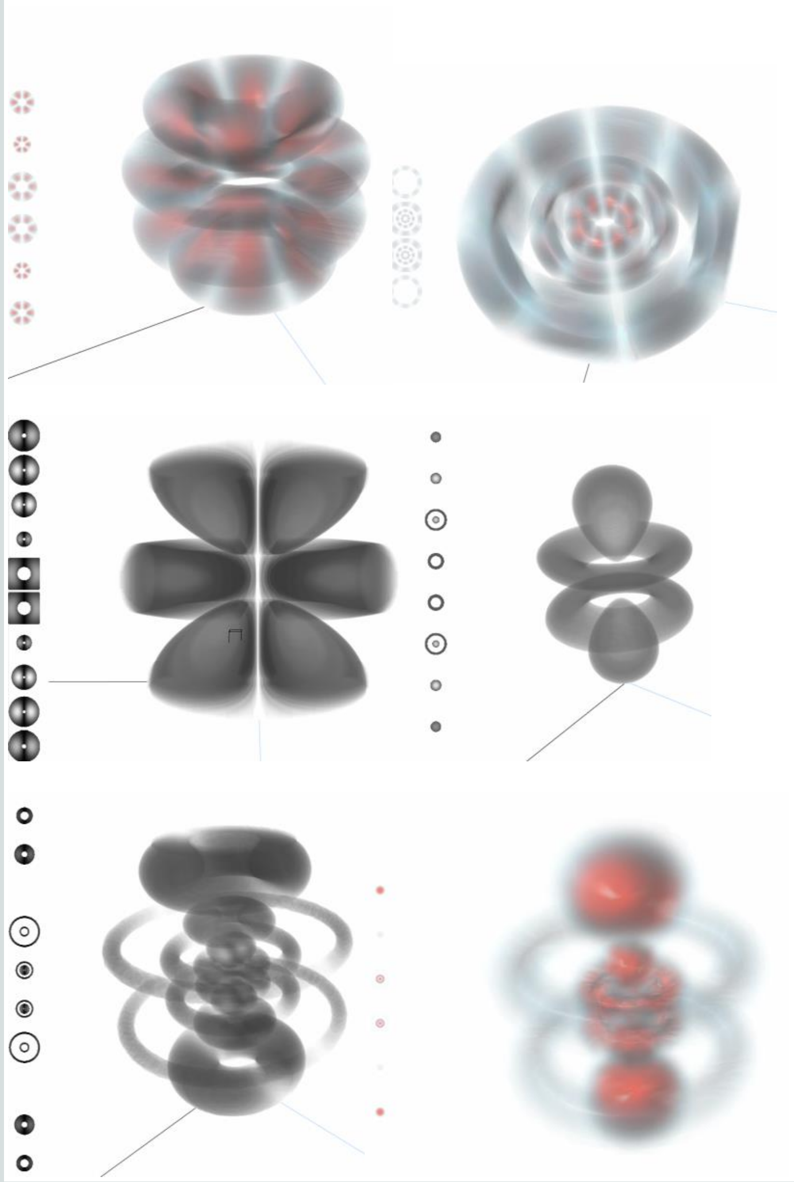
Branch Phase: controls the rotation of the new branches around the branching point

Strength: Is the strength of the generated voxel field. For example you can make a tree whose density fades with distance from the root to the leaves.

Gradient: Not used at the moment

3.1.5.5.3 Atomic orbital

The hydrogen orbital layer creates visualizations of atomic orbitals for hydrogen. These are based on spherical harmonics and create some interesting three dimensional density fields. It is also a demonstration of the type of visualizations that are possible with multi material continuum gradient techniques.



You can choose the relative size of the orbital and determine its shape by the 3 quantum numbers n, l and m.

The number n is the principal quantum number, a positive integer that determines the shell of the orbital. It is related to the mean distance of the electron from the nucleus. Geometrically it affects the size and number of spherical shells as we move away from the nucleus.

The number l is the azimuthal quantum number and its value cannot be greater than n-1;

The number m is the magnetic quantum number

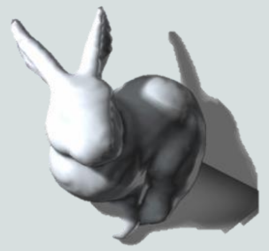
3.1.5.6 Import

3.1.5.6.1 Referenced vol file

Use this filter to insert a reference to a vol file in the layer stack. This creates an image layer that you can move rotate and scale but not edit. The content of the layer is linked to the referenced vol file. In this way you can reduce the size of models that require multiple instances of the same voxel-image to appear in different locations. If you use referenced files it is a good idea to save your lith file using the “save with resources” method. This will collect all referenced file in the same folder as the lith file and convert all references to relative file paths.

3.1.5.6.2 Import and voxelize mesh

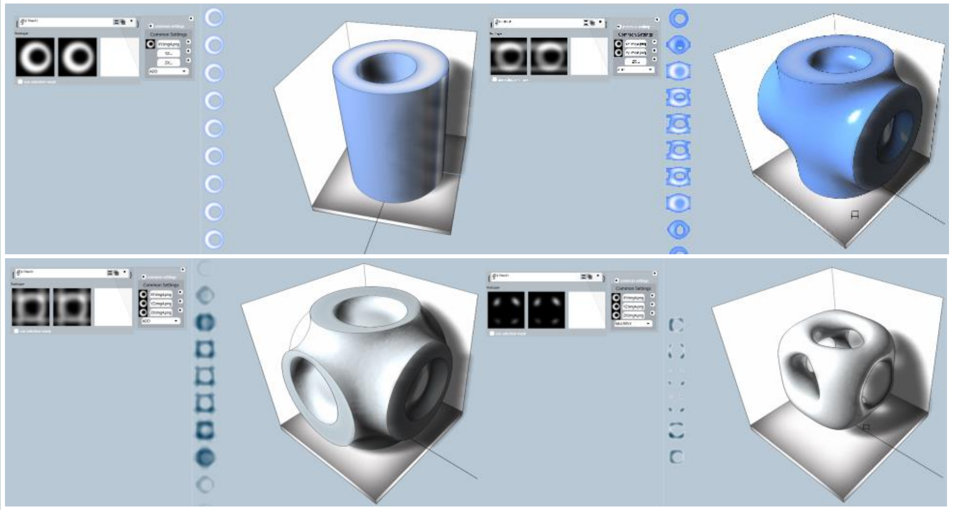
This filter will import meshes from a variety of 3d formats, and voxelize them. Meshes need to be water tight for the operation to succeed. This layer produces a voxel image with the chosen resolution along the x axis. In this image all voxels inside the mesh are assigned the “fill value” selected and all voxels outside the mesh get the value 0. The voxelization uses super-sampling so there is antialiasing along the edges of the mesh.

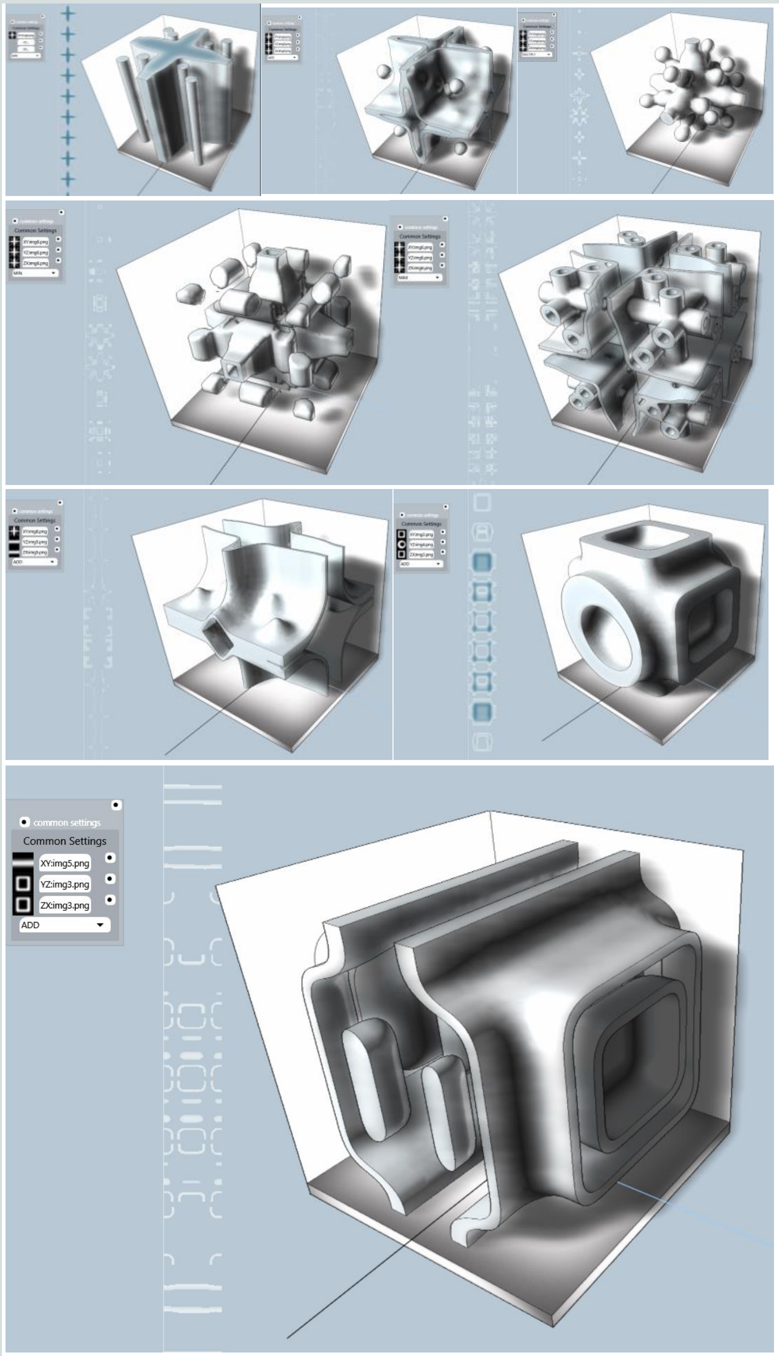


3.1.5.7 Image based

3.1.5.7.1 Image cube composite

The image cube layer extrudes and combines three images along each of the XYZ axes. The example below shows the same image extruded along 1 [top left] 2 [top right] and three axes [middle left] using different composition functions. Using composition mode “ADD” [middle left] blends the extruded images, “MULTIPLY” [middle right] creates a smooth intersection. “MAX” and “MIN” [bottom left and right] work similarly to “ADD” and “MUTLIPLY” but with sharper creases along intersections.




common settings

Common Settings

XY:img5.png

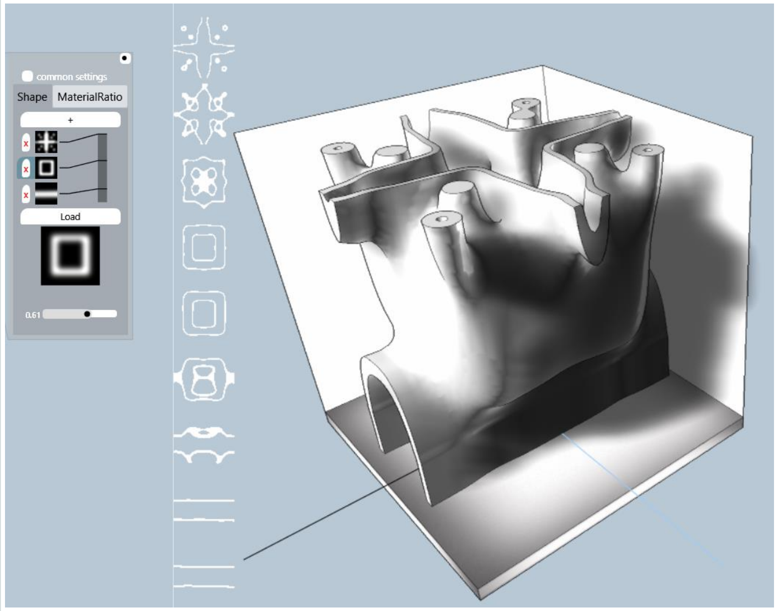
YZ:img3.png

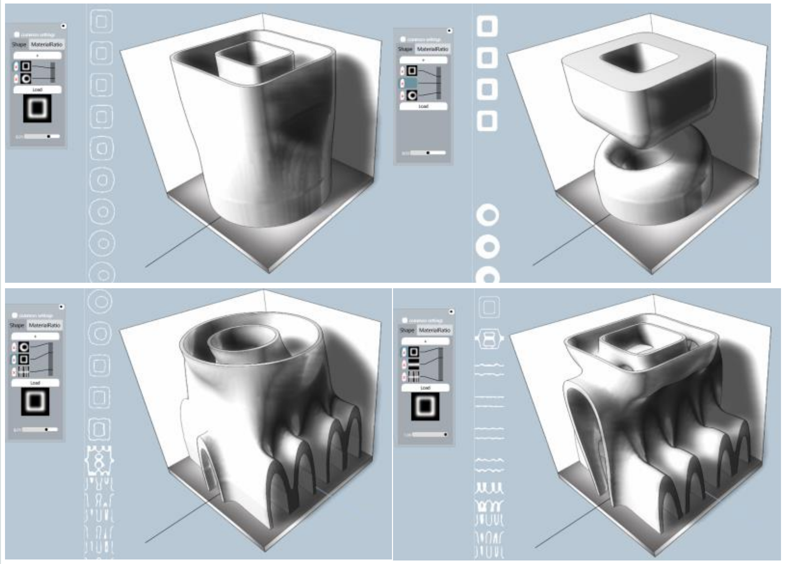
ZX:img3.png

ADD

3.1.5.7.2 Image loft

The image loft layer creates shapes that interpolate between images stacked along the z axis. You can import an unlimited number of image sections using the “+” button. Clicking on an image, highlights it and causes the image control panel to show. In this panel you can load a new image using the “load” button or change the z level of the image using the slider at the bottom.

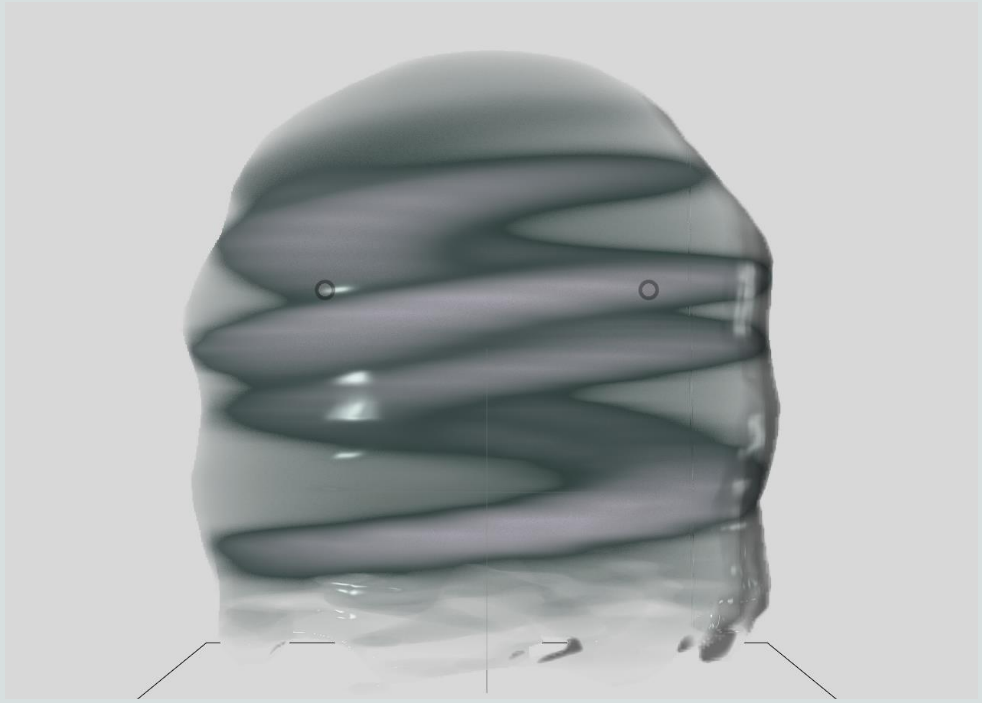


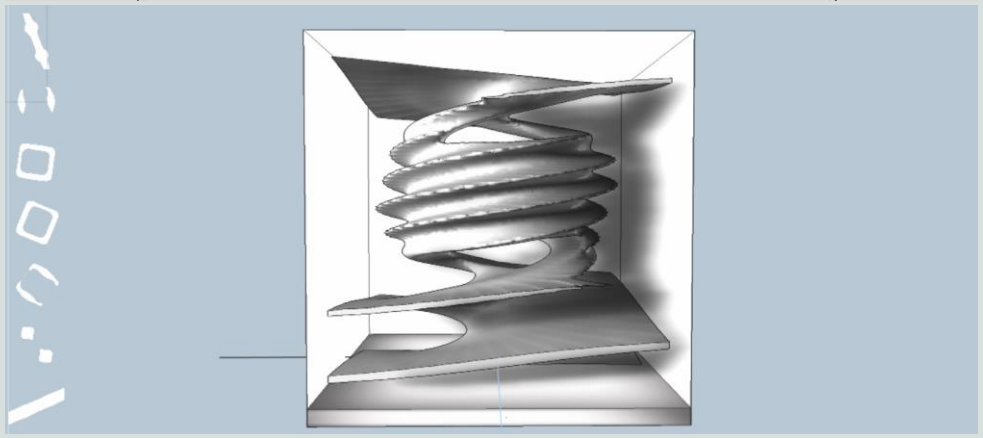


3.1.5.8 Distortion filters

3.1.5.8.1 Twist

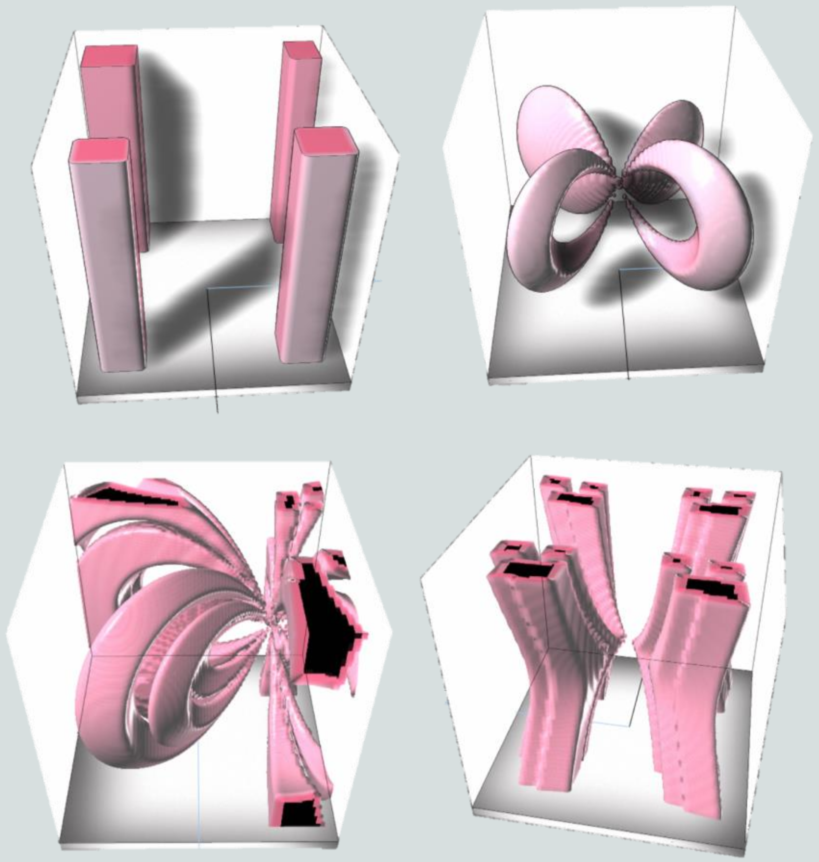
The twist layer as its name implies, applies a torsional deformation to the underlying image.





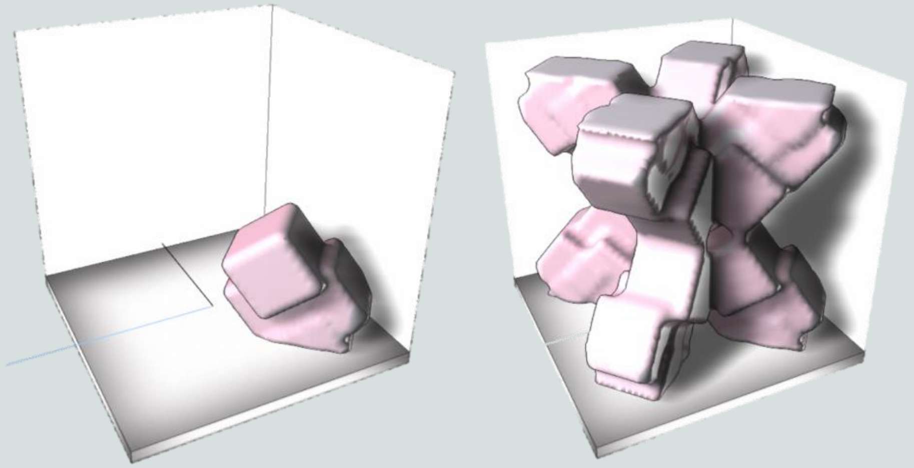
3.1.5.8.2 Inside out / inversion

The inversion filter turns the voxel image inside out around an inversion point. Points near the image boundary are mapped to the center and center points are spread on the boundary.



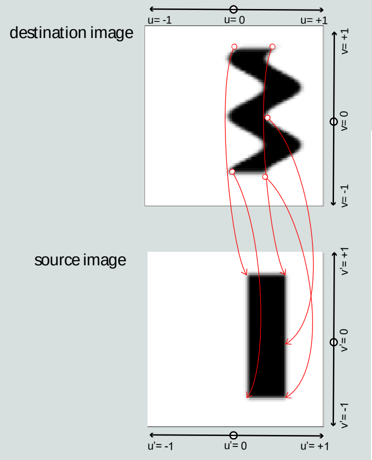
3.1.5.8.3 Symmetry

The symmetry filter imposes a multi-planar symmetry on the voxel image by mirroring and blending voxel values across 3 planes.



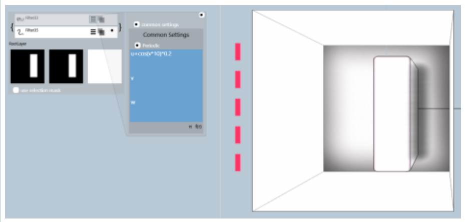
3.1.5.8.4 Function based distortion

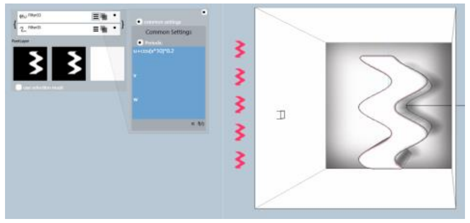
The purpose of the function based distortion filter is to enable users to apply customized non- linear deformations on the voxel image. The ui is similar to the function based field generator. Here however you need to input 3 formulas of the form:



$$u' = f_u(u,v,w)$$
$$v' = f_v(u,v,w)$$
$$w' = f_w(u,v,w)$$

The evaluator here will scan all the voxels in the voxel image. For each voxel it will calculate its normalized coordinates [u,v,w]. Then these coordinates will be passed to each one of the formulas to yield a new set of uvw coordinates [u',v',w']. The value of the voxel [u,v,w] in this layer will become equal to the value of the voxel at [u',v',w'] in the layer below. In a sense this filter establishes a mapping between the voxel spaces of the two layers. [it is important here to note that this is a "lookup" transform of the 3d texture coordinates, so if you are used to applying "active" transforms to geometry [e.g. a rotation matrix] you should think of this as applying the inverse of such a transform to achieve the same effect. This means that we don't know where each point on the source image will end up, we only define where each point in the destination image is coming from]





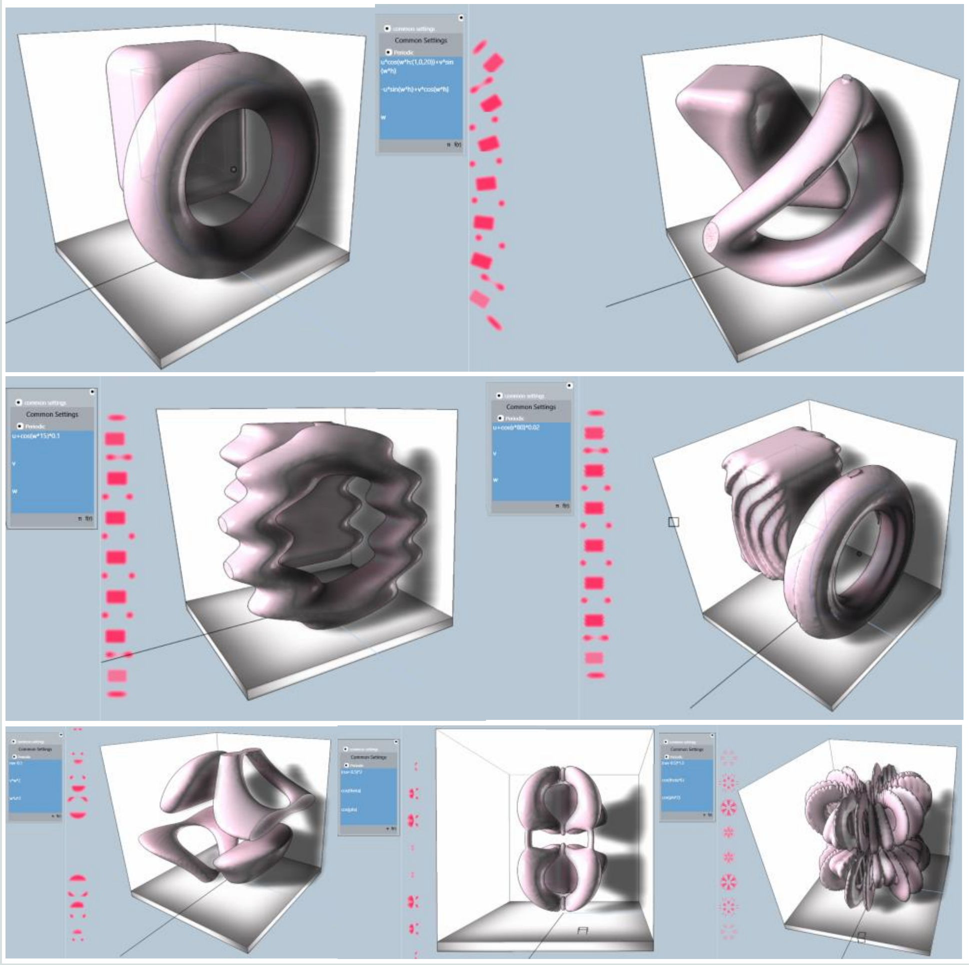
the example here demonstrates the following transformation:

$u' = u + \cos(10*v)*0.2$

$v' = v$

$w' = w$

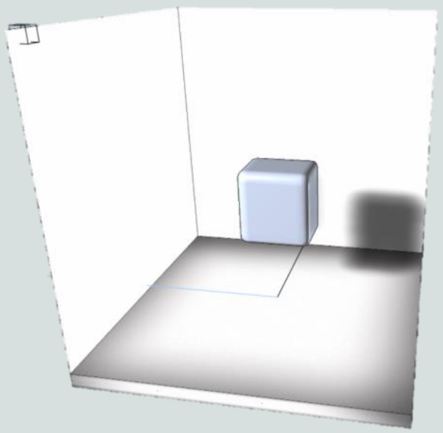
This means that along the y and z axis (v and w) there will be no distortion. The v coordinate of the destination will map to the exact same position of the source. However the u coordinate has a cosine wave that swings along the y axis [cos(10*v)].

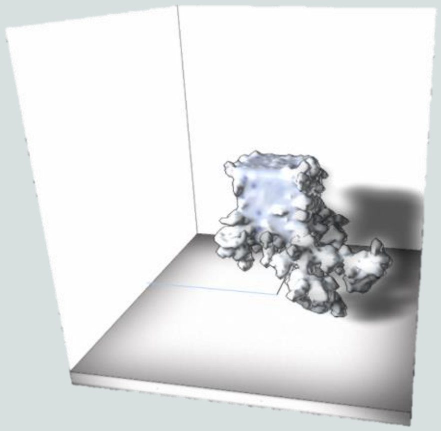


3.1.5.9 +-Noise

3.1.5.9.1 Diffusion limited aggregation

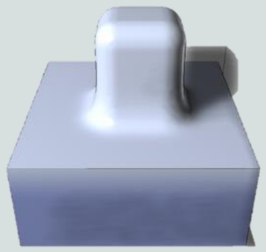
DLA is a random growth filter. It is still under development and its implementation may change significantly in later versions of monolith.

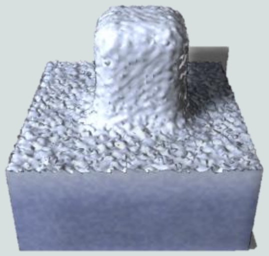




3.1.5.9.2 Noise

The noise filter as the name suggests, adds noise to the underlying voxel field.



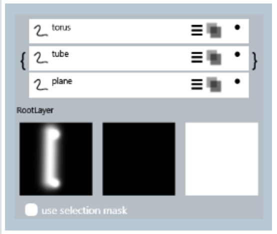


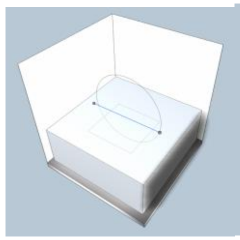
3.1.5.10 Grouping

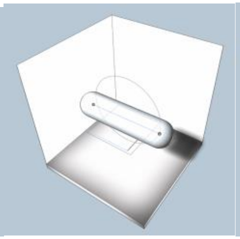
3.1.5.10.1 Layer stack

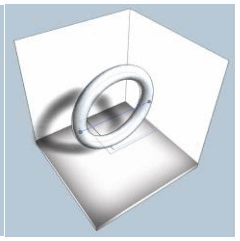
A layer stack represents a set of layers [voxel images and filters] that are composited on a temporary image before the result is composited back to the main stack. This allows for a hierarchical stacking, whereas two layers can be subtracted from each other before the result gets added to the main stack.

As an example of layer stack grouping here is a sample containing three layers. The bottom one is a box filling the bottom half of the image, the middle layer contains a tube and the top layer a ring.

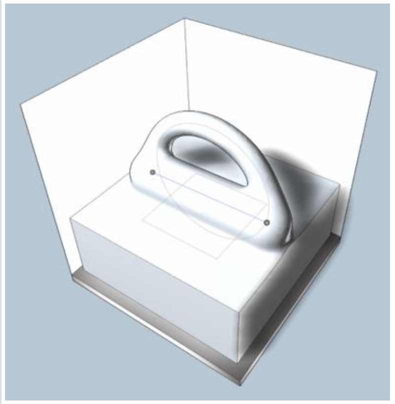




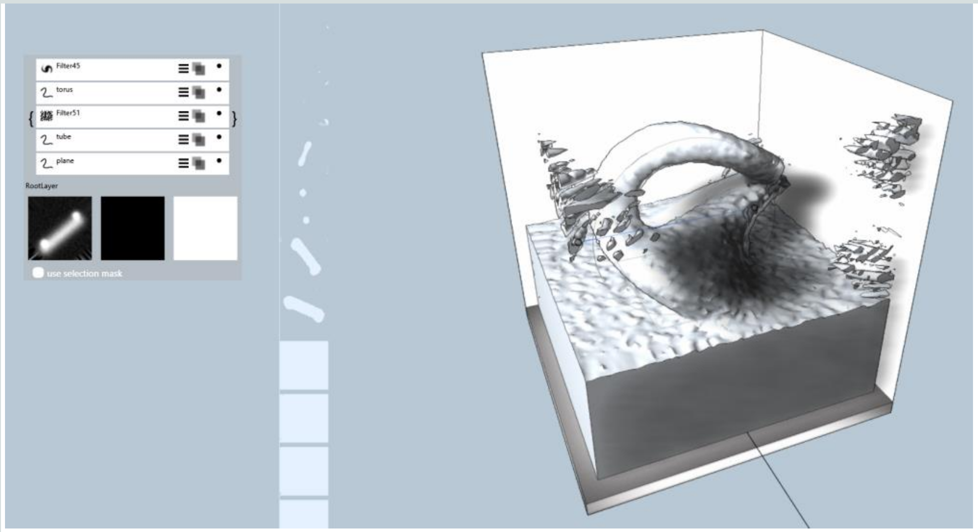




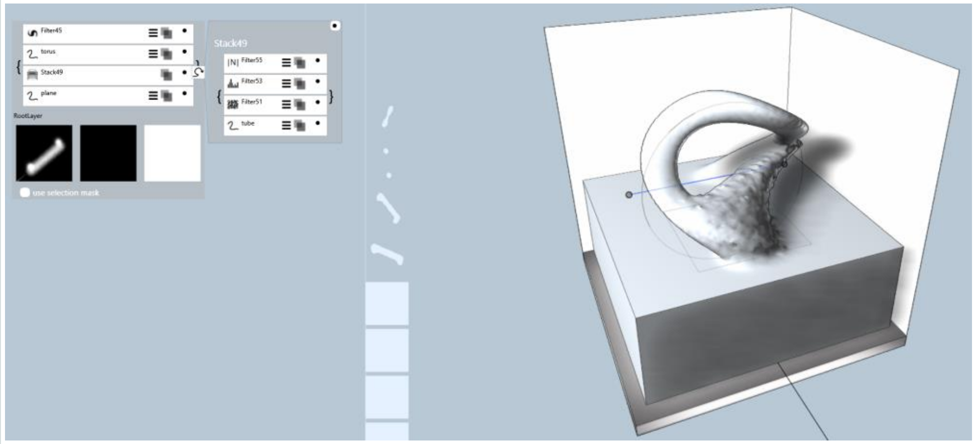
The result of stacking the three layers is a composite object as shown in the following image:



Now assuming that we wanted to apply some noise on the tube and twist the torus without affecting the whole image. If we just add the noise and twist effects we will get the following:

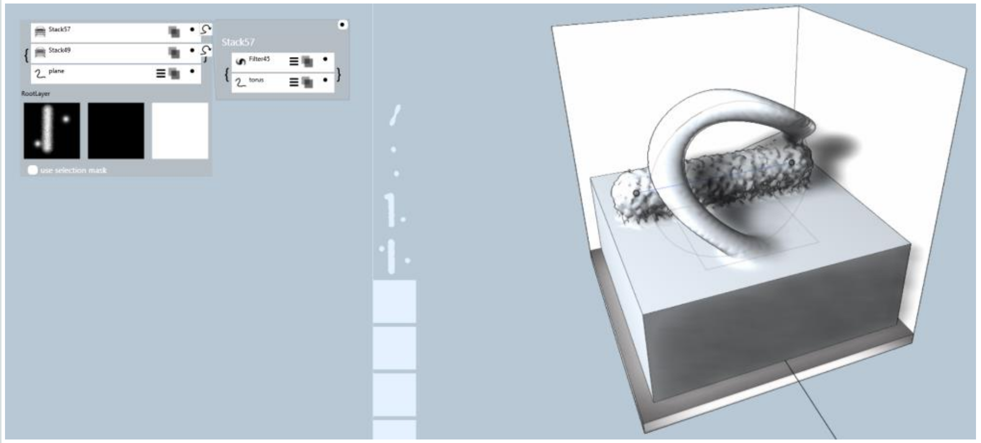


The noise is applied to the tube as we wanted but also to the plane surface since it is just composited on top of the [plane + tube] image. The twist is also applied to everything and not just the torus. In order to isolate the effects we can use stack layers:



First placing the tube and the noise [plus some histogram and renormalization filters to remove noise from the black regions] inside a stack layers we see the noise effect isolated only on the tube.

We can also group the twist effect with the torus:



So now the twist is applied to the torus only, and it is the twisted torus that is composited to the main stack.

In general use grouping stacks to isolate operations, be it filters or Booleans before merging them to the root stack.

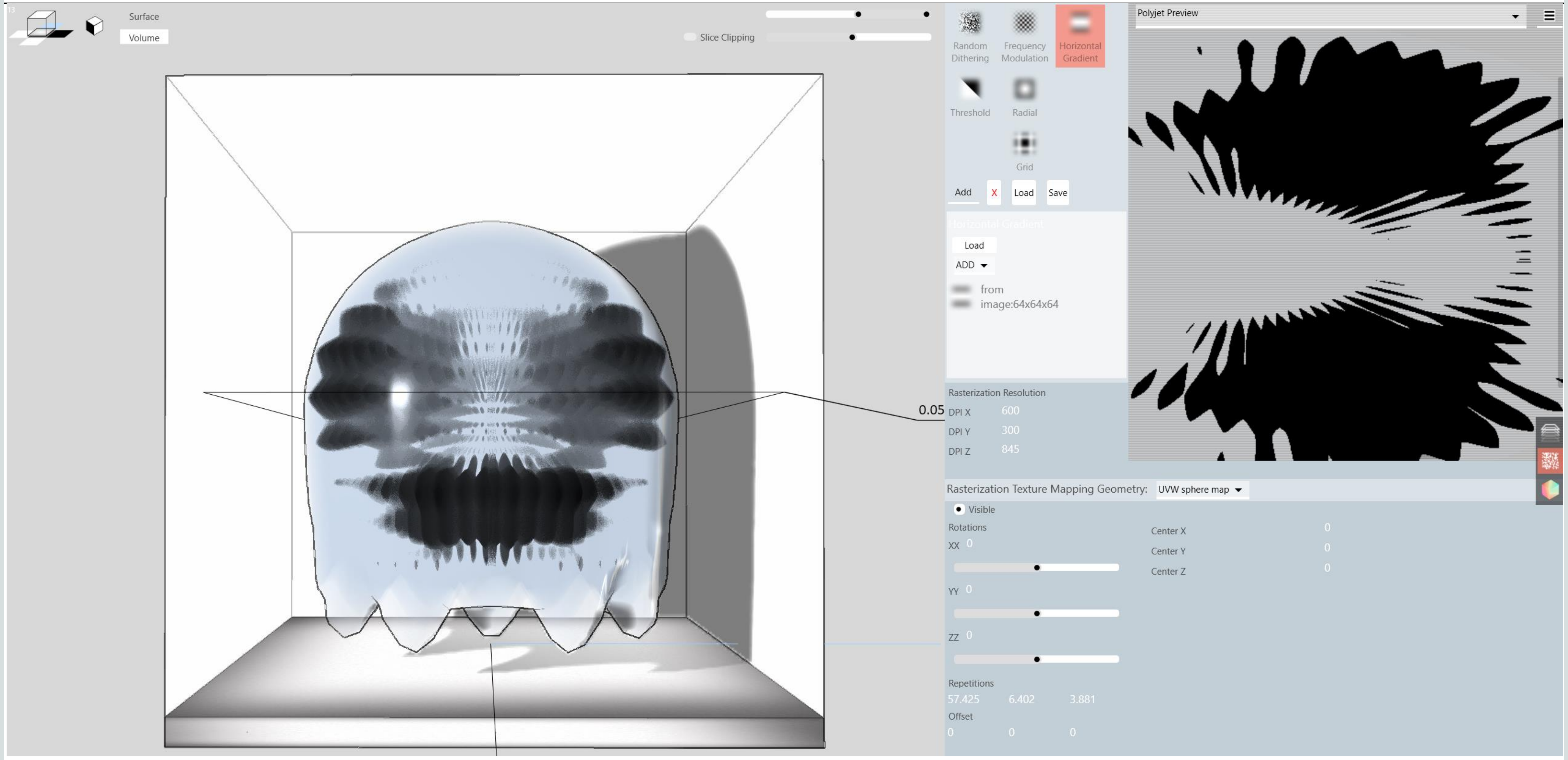
3.1.5.10.2 Layer stack with affine transform

The layer stack with affine transform works similar to the simple layer stack with the difference that the contents of this stack can be moved scaled and rotated as a group. Using stacks with transformations is computationally more expensive than the simple stack so if you don't need to apply any transforms use the regular stack when possible.

See the section on layer transformations that explains the transformation panel.

3.2 Rasterization workflow

3.2.1.1 The rasterization User interface



Rasterization Resolution

DPI X	600
DPI Y	300
DPI Z	845

Rasterization Texture Mapping Geometry: UVW sphere map

Rotations
XX 0
YY 0
ZZ 0

Center X	0
Center Y	0
Center Z	0

Repetitions
57.425 6.402 3.881
Offset
0 0 0

When you select the top level workflow “rasterization” [buttons on the far right edge] the layer compositing interface is replaced by the rasterization controls.

This panel has three major regions.

On the top left is the pattern selection and definition area where you can select or define your own rasterization patterns.

On the top right there is a preview window on which you can inspect a horizontal slice through the model in various ways, including the exact high resolution pattern that would be sent to a printer

Finally on the bottom half of the panel are the texture mapping controls the will determine the local orientation and flow of the pattern as it is applied to the voxel image.

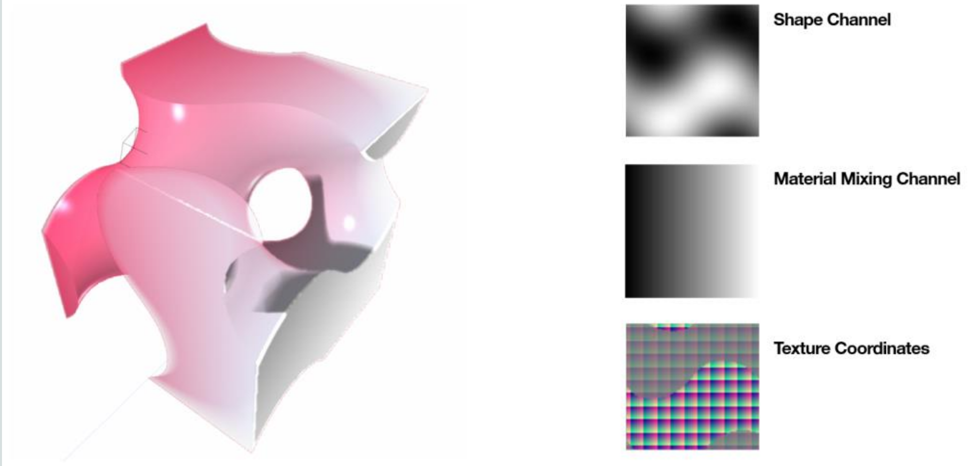
3.2.1.2 The patterning process

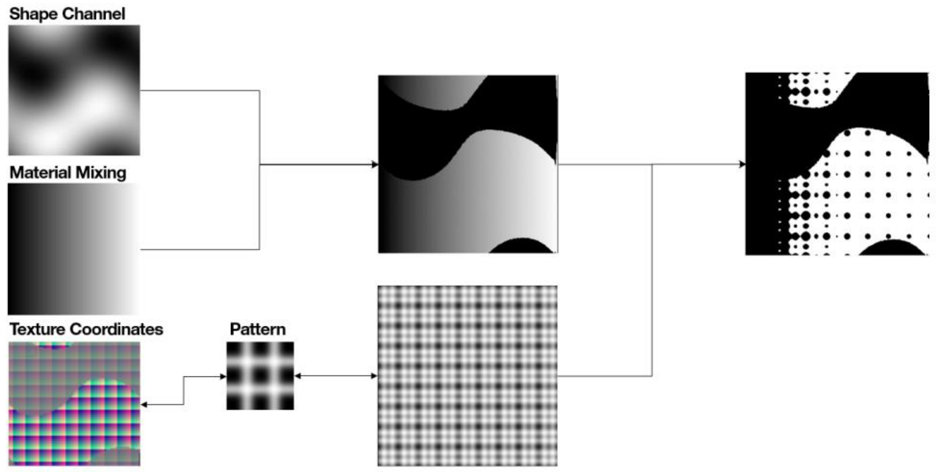
There are three elements that will determine the rasterization pattern in an image:

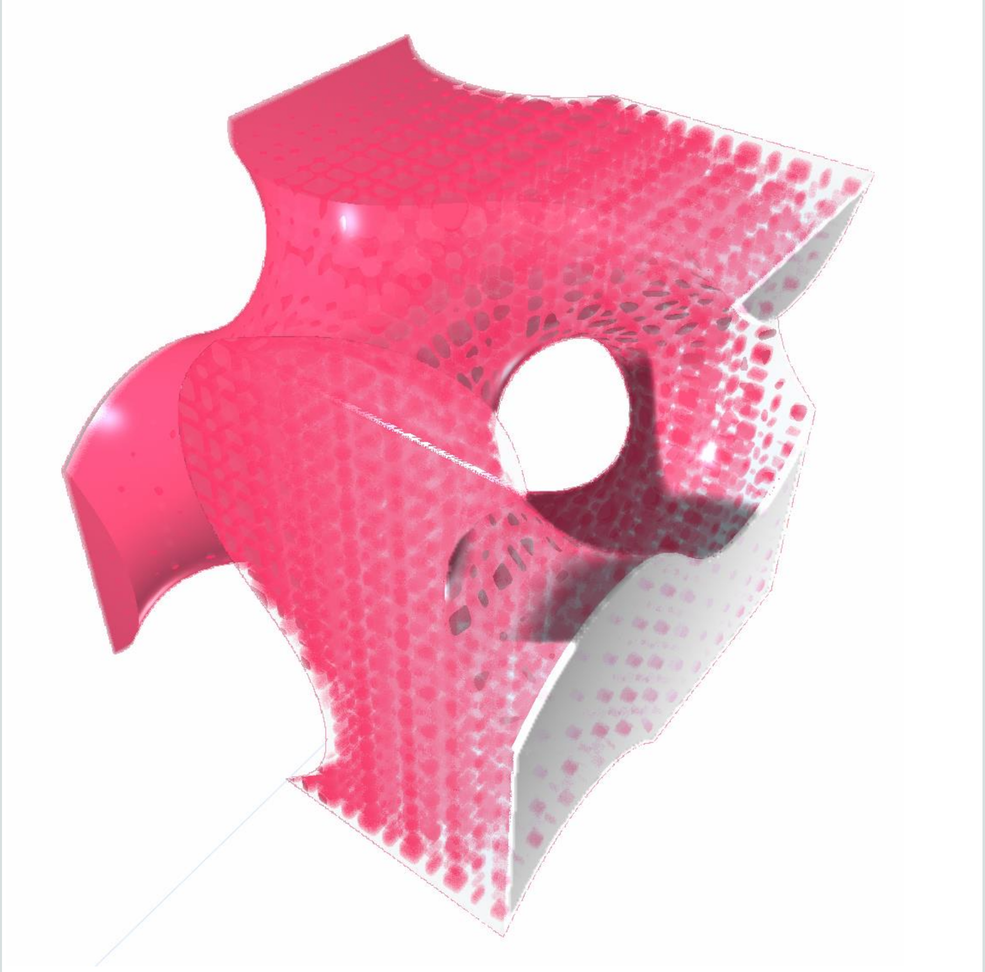
First the material gradient channel will determine the porosity of the pattern [where to place more of one material than the other]

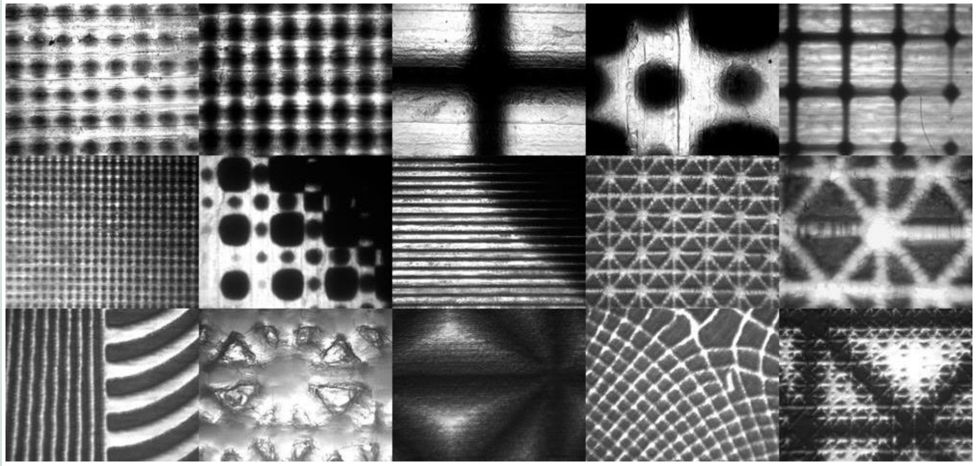
Second the selected raster pattern will determine the shape of the "dots" or other micro structures that realize this porosity

Third the uvw texture coordinates will determine the flow of the pattern so that you can create highly controlled anisotropic materials.



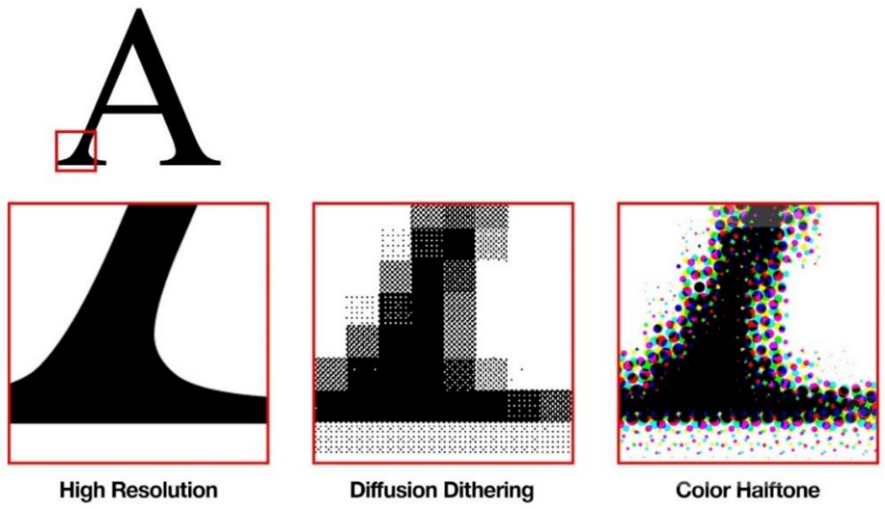






3.2.1.3 Halftone Patterns

Halftoning or screening is a printing process which originated out of the need to reproduce continuous tone imagery which contained an infinite range of colours or greys using only one colour of ink and dots of different sizes. Given that the printed dots are quite small, the reproduction process relies on an optical illusion where the human eye blends the halftone dots into continuous smooth tones. New techniques have emerged through advancements in digital technology to reproduce high resolution vector graphics (displayed on screen) into finite resolution (ie. 600dpi) printable images.

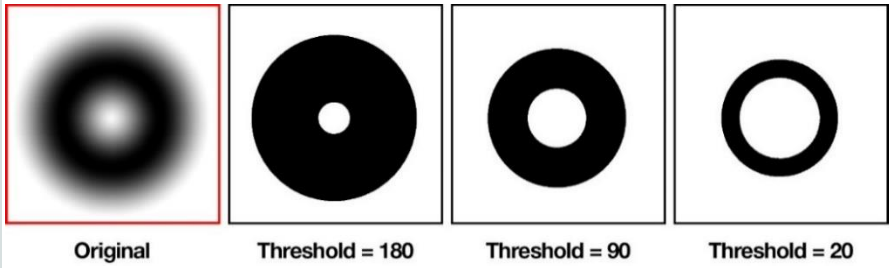


Monolith faces a similar problem where we need to translate a 3d voxel image into a format compatible with a finite-resolution printer – only now we need to use processes which can work in three dimensions. Fortunately, we can draw upon existing digital image processing techniques as a starting point. Dithering is one of the most common techniques used because of its ability to approximate continuous grey-scale values out of discrete black and white elements.

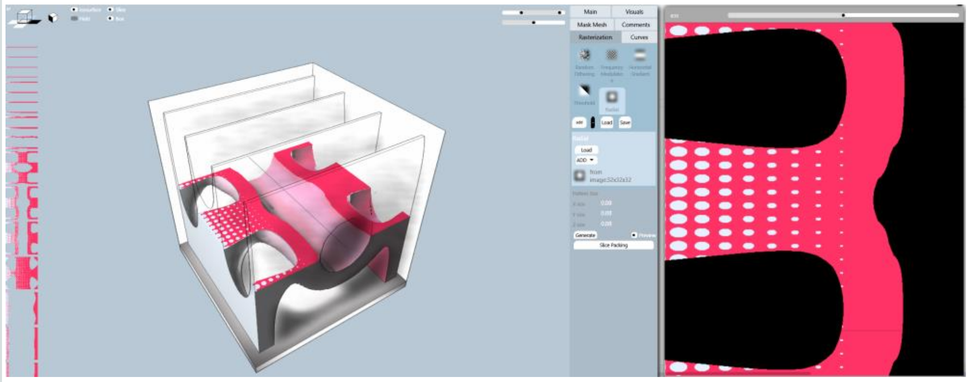
Monolith's default rasterization pattern is a *Random Dithering* which is a stochastic process which attempts to approximate a continuous gradient using a limited color palette known as color quantization. Dithering, by its nature, introduces pattern into an image. For the most part, these patterns are not discernible by the human eye because a single pixel in the raster pattern (when viewed at full-scale) corresponds to a single drop of resin on the 3d printer which is on the order of micrometers. Nevertheless, the random dithering pattern introduces an element of stochasticity to perturb the algorithm a little.

The random dithering pattern is macroscopically isotropic and works best if you want a very smooth continuous gradient between two materials. However, the drawback is that the images that are output (one for each slice layer in the 3d print) are not readily compressible. Compression algorithms work by approximating contiguous regions within an image which contain the same color value. The random nature of this pattern precludes this from being effective.

A more simplified form of dithering is known as *Thresholding* where each value is compared to a fixed threshold value. In this algorithm, all pixels whose intensity level lies above the threshold value are quantized to 1; all others get a value of 0. The following diagram shows this process applied to a two-dimensional image, however it can easily be applied to a three-dimensional voxel field as well.



By using trilinear interpolation, we can determine the density of material at any location within the boundary volume. In many instances, the voxel grid resolution will be lower than the printer's native one so the rasterization process has to interpolate between values. Like most interpolation methods, this process estimates values that fall within range of known data points – in this case the eight corner points of each voxel cube element. Then at each pixel, we use a tri-linear interpolation of the object fields to find the value at the corresponding point as well as a trilinear interpolation of the halftone field. If the interpolated density value is greater than the threshold then the pixel is a solid otherwise it is void. If the pixel is solid we check to see if the interpolated material mixing value is greater than the interpolated halftone value and the result of this check determines whether the pixel is assigned one material or the other.

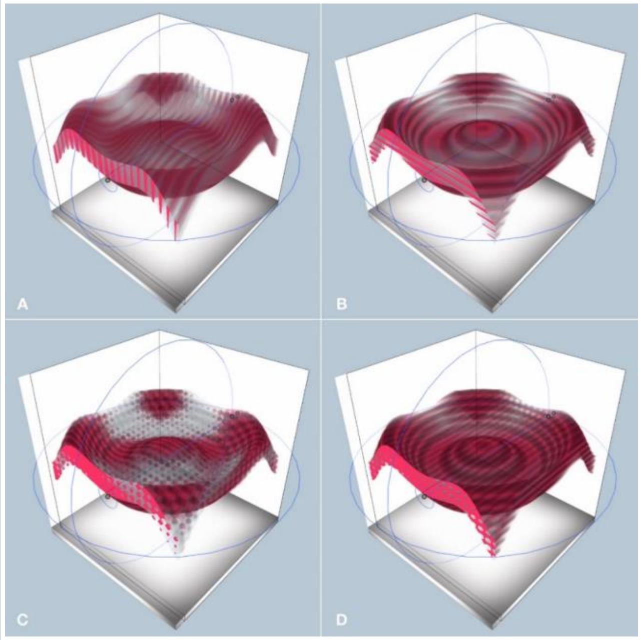


3.2.1.4 Raster from Image

In addition to dithering, we can use other halftone techniques. In halftoning, we use patterns to approximate continuous tonal gradients. Those patterns can be simple dots or lines (each of which are included under the rasterization tab) or through custom 2D images or 3D voxel patterns.

To create a halftone pattern from an image, simply create a black and white image using your favourite image editing software application. While it is possible to use larger images as a halftone pattern, it is often a good idea to keep your images under 64px wide by 64px high to reduce your memory footprint. In Monolith, click on the “add” button under the rasterization tab. The drop-down menu will ask you to specify whether you want to define a pattern using a 2D bitmap or a 3D voxel pattern. After choosing a bitmap pattern, a blank template pattern will be inserted next to the existing patterns. By selecting that pattern in the menu, you will be able to change its attributes like its name, pattern size, and the bitmap image you would like to use.

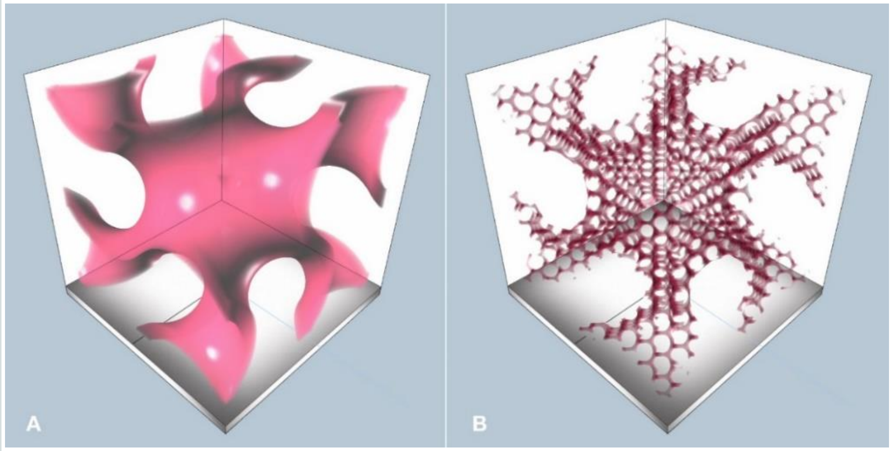
Once you have a bitmap loaded into a pattern, you can specify how you want that image mapped throughout the volume. Mapping modes can have a dramatic visual impact on your model and can be used to enforce optical or structural anisotropy at the rasterization scale. Current mapping methods include extruding along the XY, YZ, or XZ axes as well as multiplicative or additive blending modes.



Mapping modes using a horizontal linear gradient bitmap: [A] Extrude XY [B] Extrude YZ [C] Additive Blending [D] Multiplicative Blending.

3.2.1.5 Raster from Voxels

Similar to raster patterns from images, Monolith can create halftone patterns from existing 3d voxel images. To create a voxel halftone pattern, simply click on the add button under the rasterization tab and choose from Voxel Pattern. A blank template will be added to the rasterization menu. By selecting this new pattern, you can select which voxel pattern you would like to use. The From Current button will load the current voxel image as the propagating pattern. The load button will allow you to select an existing .vol file to use as the underlying pattern.



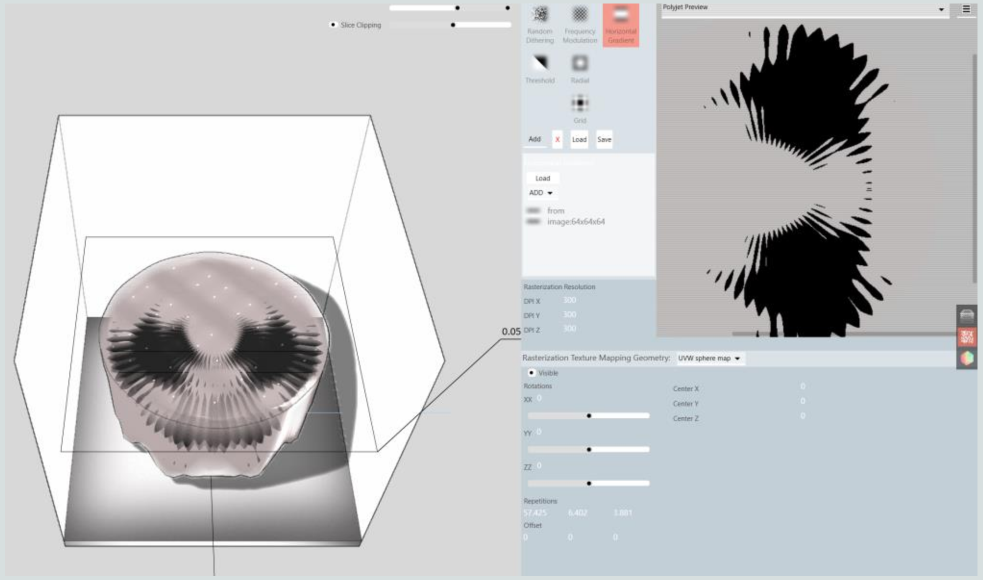
[A] 3d voxel image of a scalar gyroid function and that same gyroid pattern used as a mesocopic halftone [B].

3.2.1.6 Preview Rasters in 2D and 3D

Many multimaterial 3d printers include software applications which prepare existing files (usually meshes) for the printer. For ink-jet printers, this software will create a series of bitmap images – one for each print head corresponding to a single pass over the print bed. We often refer to these bitmaps as slices. In slices, a white pixel tells the printer that the corresponding print head should add a drop of resin while a black pixel indicates no resin should be added.

One of the benefits of working within a voxel field – which is analogous to a pixel space – is that we can generate these slice files directly from the existing data structure. For each printed layer, the algorithm generates two monochromatic bitmaps - one for each material using the resolution specifications of the 3D printer. Then at each pixel, we use a tri-linear interpolation of the object fields to find the value at the corresponding point as well as a trilinear interpolation of the halftone field.

We can view a 2d composite preview of the raster slice file by selecting the Polyjet *Preview* option at the top of the preview window. This will show a full-scale preview of the slice file. Full-scale means that every pixel in the image corresponds to a single drop of resin in the 3d printer. The preview image may appear stretched or skewed when compared to the 3d counterpart, depending on the printer resolution that you choose.



Rasterization Resolution
DPI X 300
DPI Y 300
DPI Z 300

Rasterization Texture Mapping Geometry: UVW sphere map

● Visible
Rotations
XX 0
YY 0
ZZ 0
Center X 0
Center Y 0
Center Z 0
Repetitions
57.425 6.402 3.881
Offset
0 0 0

3.2.1.7 Exporting rasterized slices for printing

The 2d preview is actually a composite of the two materials. You can save the current slice or all the slices by using the options in the properties menu above the preview ☰

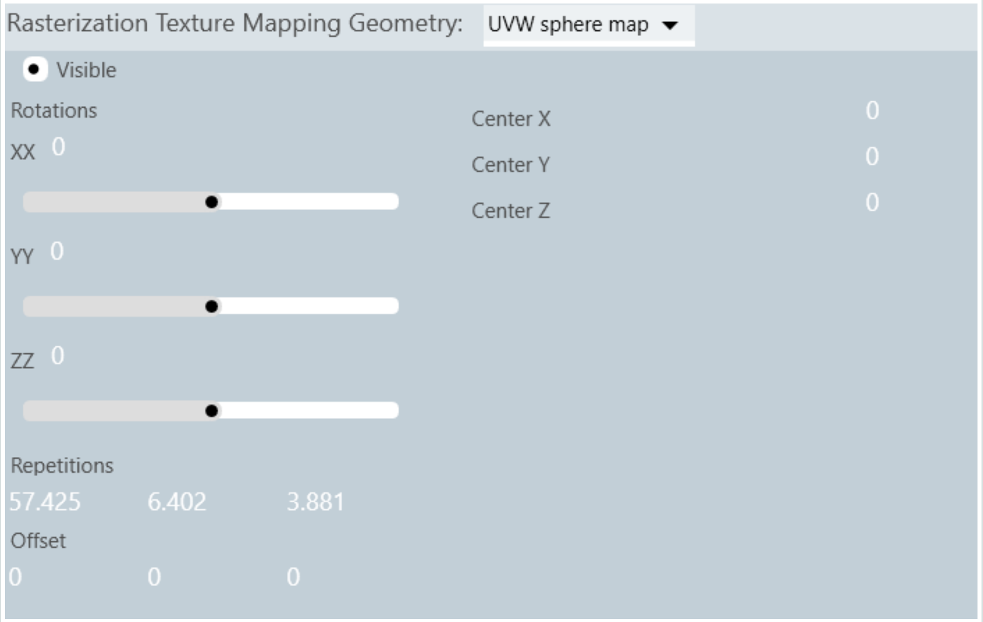
Exporting all the slices to high resolution will save a series of images [either pairs of 1 bit per pixel BMP or 8bit or 32 bit png files.

This operation can be time consuming but runs on a separate thread so you can continue working while this is going on. A progress window will pop up as soon as you initiate one of these export jobs.

3.2.1.8 Raster mapping and flow: the three dimensional texture coordinates

As mentioned before you can control the shape of the rasterization pattern and its porosity by the material gradient channel. In addition you can define how the micro pattern will flow through the volume using a texture map. This is very similar to texture mapping techniques used in 2d surface modelling software however the texture coordinates now have one more component that determines the propagation of the voxel pattern within the thickness of a solid surface.

The user interface for defining the texture mapping is right below the raster selection and preview panels:



Rasterization Texture Mapping Geometry: UVW sphere map

● Visible

Rotations

XX 0

YY 0

ZZ 0

Center X 0

Center Y 0

Center Z 0

Repetitions

57.425 6.402 3.881

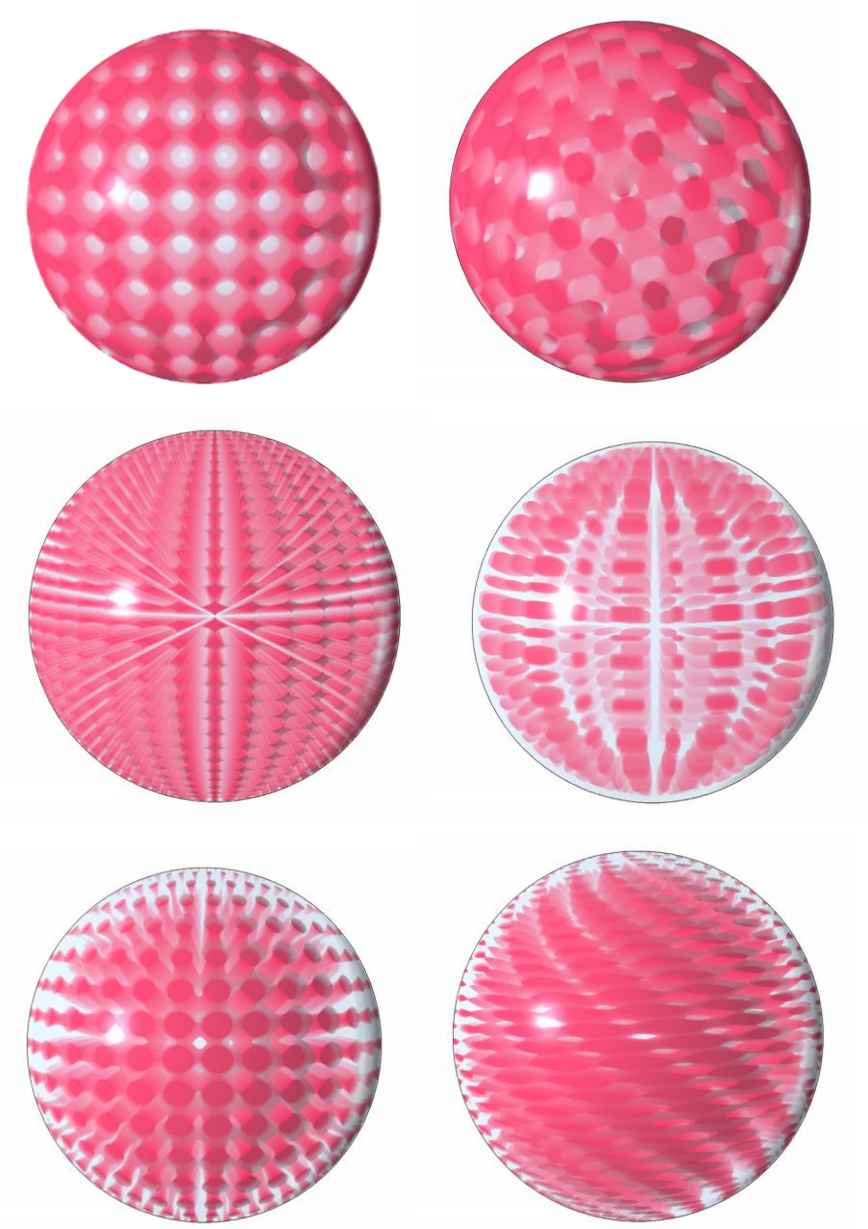
Offset

0 0 0

The combo box at the top enables you to select the type of parametric map to use. It contains the typical maps [box, spherical, cylindrical etc…] plus a special mesh based flow. All texture maps have the left side column of controls that determines the uniform orientation and scaling of the pattern, which corresponds to repetitions.

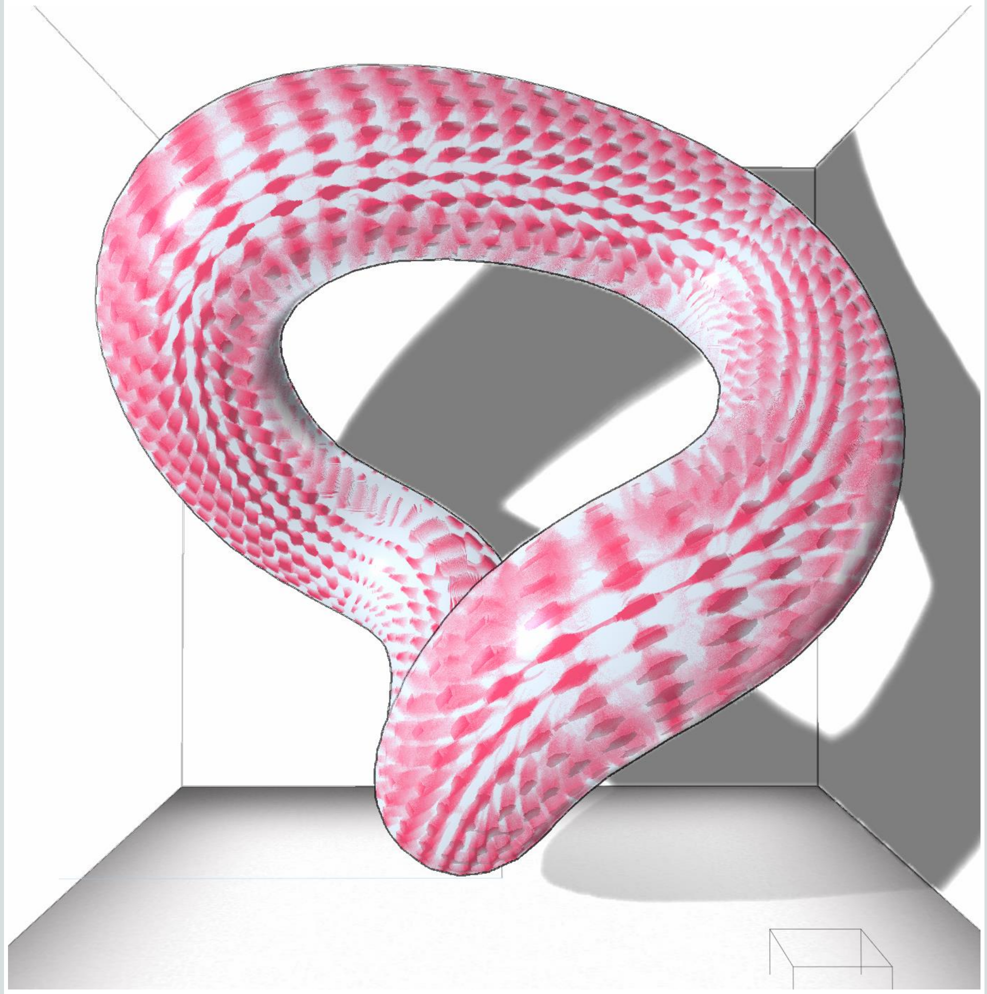
The right side controls are determined by the specific map you choose.

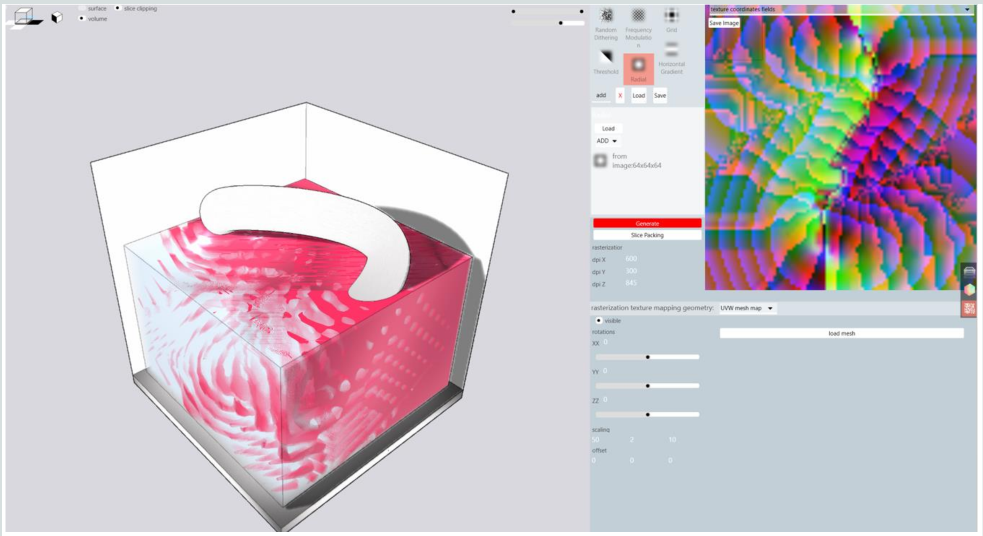
Here are some examples of the typical texture maps and their effect on a spherical object:



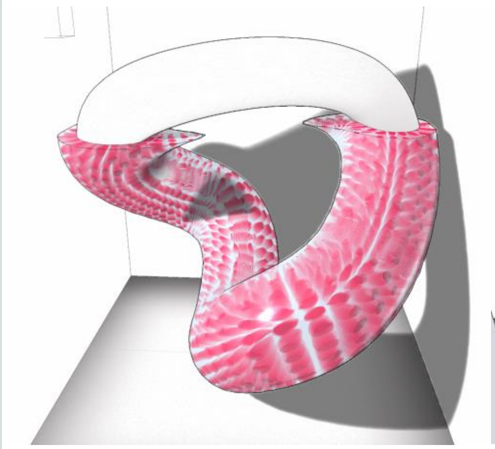
3.2.1.8.1 The mesh flow texture map

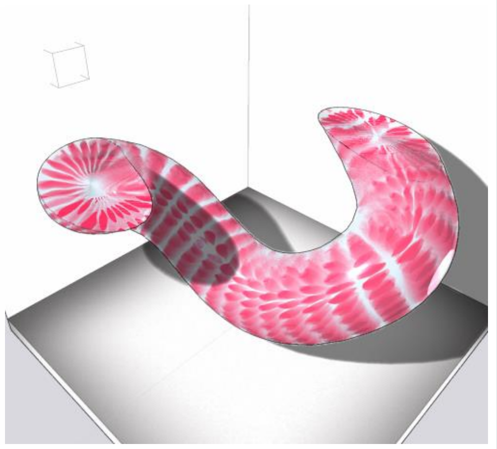
In addition to the above uniform texture maps there is a mesh based map that will enable you to propagate the volumetric texture coordinates along any geometry. This map will use the UV coordinates from a surface model that you import and then propagate them inwards and outwards from the surface in order to generate the third volumetric coordinate. You can use this method when you select the "UVW mesh map" from the texture mapping selection combo box.



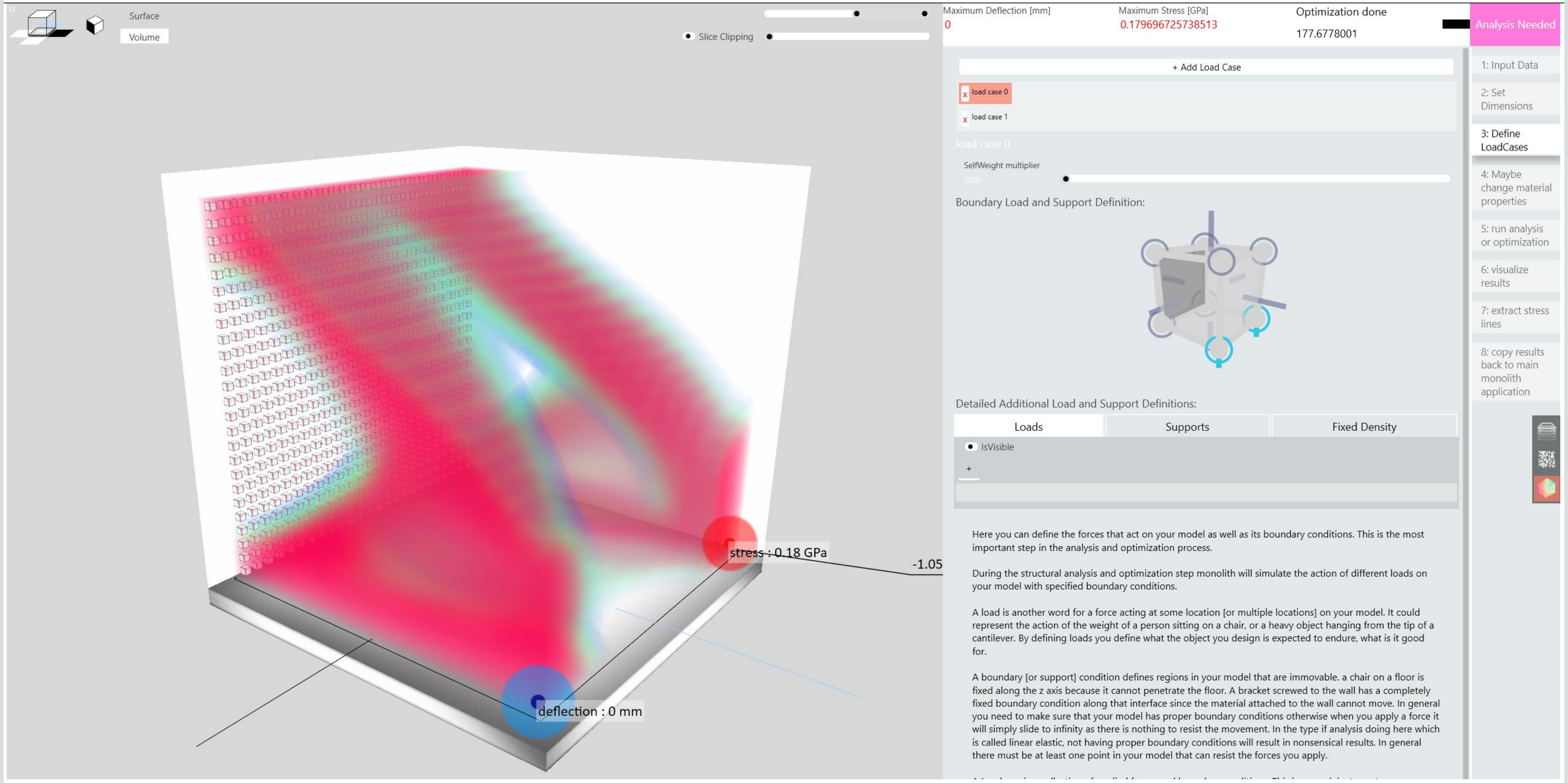


rasterization texture mapping geometry: UVW mesh map





3.3 Structural Analysis workflow



Here you can define the forces that act on your model as well as its boundary conditions. This is the most important step in the analysis and optimization process.

During the structural analysis and optimization step monolith will simulate the action of different loads on your model with specified boundary conditions.

A load is another word for a force acting at some location [or multiple locations] on your model. It could represent the action of the weight of a person sitting on a chair, or a heavy object hanging from the tip of a cantilever. By defining loads you define what the object you design is expected to endure, what is it good for.

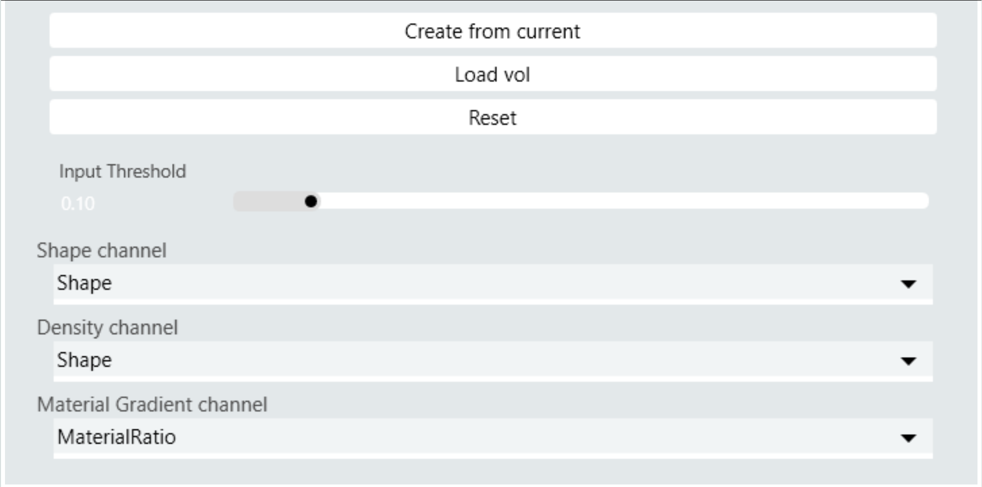
A boundary [or support] condition defines regions in your model that are immovable. a chair on a floor is fixed along the z axis because it cannot penetrate the floor. A bracket screwed to the wall has a completely fixed boundary condition along that interface since the material attached to the wall cannot move. In general you need to make sure that your model has proper boundary conditions otherwise when you apply a force it will simply slide to infinity as there is nothing to resist the movement. In the type if analysis doing here which is called linear elastic, not having proper boundary conditions will result in nonsensical results. In general there must be at least one point in your model that can resist the forces you apply.

The final top level workflow exposes the structural analysis and optimization functionality in monolith. This workflow will enable you to run simulations and test the strength and distribution of stresses within the volumetric model. In addition a topology optimization module can help you determine lightweight minimum compliance structures that meet certain requirements.

The user interface is organized as a series of steps [far right column] with embedded instructions. At the top right corner there is a button that turns pink each time the model is out of sync with the results [for example when you changed some material property or boundary condition]. You can click the "analysis needed" button at any point to refresh the model. Depending on the resolution of your model this may be a time consuming actions.

You do not have to go through all the steps in order to setup an analysis and optimization model. For most cases the default options should be sufficient. The most important step is, step number 3 where you define the actual forces and boundary conditions that determine the load case you want to test or optimize against.

3.3.1 Structures: step 1: Input Data



Input Threshold 0.10

Shape channel: Shape

Density channel: Shape

Material Gradient channel: MaterialRatio

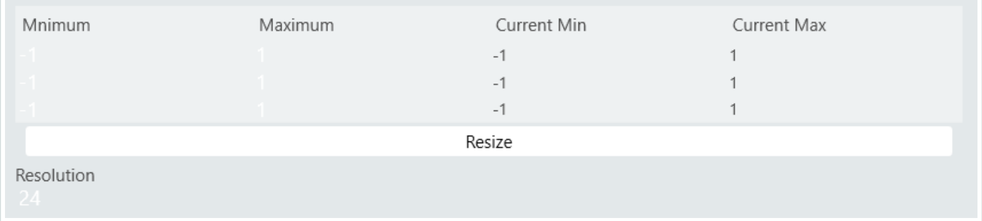
In this panel you can select the input geometry for the structural analysis module.

This module manages its own voxel model and the changes you make to it do not reflect on your main model until you output them as a layer [last tab in this workflow].

You can input a voxel image here, either by grabbing a snapshot of the current model in monolith or importing a vol file. The data of the imported voxel model will be used to decide the solid/void condition for the structural model as well as the mixing of materials within this model.

Alternatively you can skip this step and start with a fully solid box. In this case you can apply loads and support conditions to this model and use the optimization module in order to find a lightweight structure that you can then output back to the main application.

3.3.2 Structures: step 2: Set Dimensions



Mnimum	Maximum	Current Min	Current Max
-1	1	-1	1
-1	1	-1	1
-1	1	-1	1

Resolution 24

In this panel you can define the dimensions and resolution of the analysis model.

These settings will apply only on the structural model and not in the model you are creating in the main application.

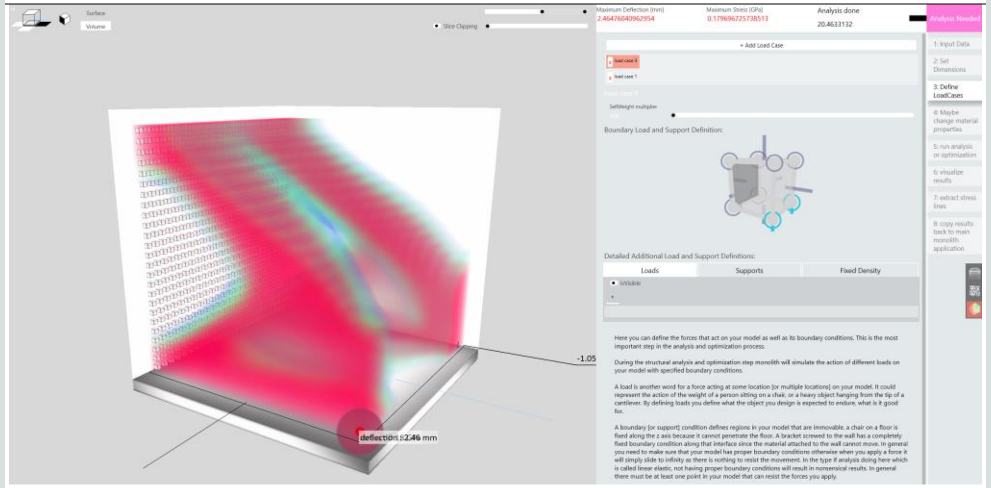
The resolution is an integer number that determines the number of voxels along the longest dimension of the box.

In general higher resolution will require more memory and more time for analysis and because of the volumetric nature of the model increasing the dimension a factor will increase the memory footprint by the same factor raised to the 3rd power and the analysis time by even more. In general start with a low resolution [about 16] where you can have near interactive rates of analysis in order to make sure loads and boundary conditions worked as you want.

At the end you can ramp up the resolution here and let the analysis or optimization run again.

Higher resolution will resolve finer structures during the optimization process and perhaps create more intricate geometries.

3.3.3 Structures: step 3: Define Load Cases



Maximum Deflection [mm] 2.46476040962954

Maximum Stress [GPa] 0.179696725738513

Analysis done 20.4633132

Here you can define the forces that act on your model as well as its boundary conditions. This is the most important step in the analysis and optimization process.

During the structural analysis and optimization step monolith will simulate the action of different loads on your model with specified boundary conditions.

A load is another word for a force acting at some location [or multiple locations] on your model. It could represent the action of the weight of a person sitting on a chair, or a heavy object hanging from the tip of a cantilever. By defining loads you define what the object you design is expected to endure, what is it good for.

A boundary [or support] condition defines regions in your model that are immovable. a chair on a floor is fixed along the z axis because it cannot penetrate the floor. A bracket screwed to the wall has a completely fixed boundary condition along that interface since the material attached to the wall cannot move. In general you need to make sure that your model has proper boundary conditions otherwise when you apply a force it will simply slide to infinity as there is nothing to resist the movement. In the type if analysis doing here which is called linear elastic, not having proper boundary conditions will result in nonsensical results. In general there must be at least one point in your model that can resist the forces you apply.

This is the most important panel in defining the analysis model. Here you can create load-cases [collections of load and support definitions].

By default a load-case is already defined [you can add more using the “add load case” button] but it contains no loads or supports.

During the structural analysis and optimization step monolith will simulate the action of different loads on your model with specified boundary conditions.

A load is another word for a force acting at some location [or multiple locations] on your model. It could represent the action of the weight of a person sitting on a chair, or a heavy object hanging from the tip of a cantilever. By defining loads you define what the object you design is expected to endure, what is it good for.

A boundary [or support] condition defines regions in your model that are immovable. A chair on a floor is fixed along the z axis because it cannot penetrate the floor. A bracket screwed to the wall has a completely fixed boundary condition along that interface since the material attached to the wall cannot move. In general you need to make sure that your model has proper boundary conditions otherwise when you apply a force it will simply slide to infinity as there is nothing to resist the movement. In the type if analysis doing here which is called linear elastic, not having proper boundary conditions will result in nonsensical results. In general there must be at least one point in your model that can resist the forces you apply.

A Loadcase is a collection of applied forces and boundary conditions. This is a convenient way to group actions together. For example you could have a loadcase representing a person sitting on a chair and a different loadcase representing a person leaning back. For a building you could have one loadcase for each direction of wind that you want to test against.

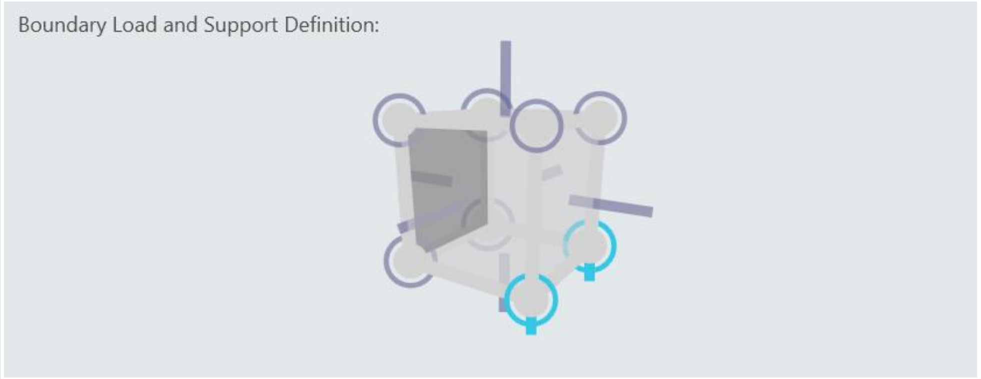
Loadcase in monolith are constructed by defining regions in space [boxes, spheres, etc...] that determine the properties of enclosed or neighbouring voxels.

All units are SI [N, m, sec ...]

You can select the current active load-case from the list. The loads and supports defined in the selected load-case are automatically assigned to the analysis system.

The self-weight multiplier determines the factor by which the self-weight of each voxel is multiplied before it is converted into nodal loads. During optimization it is best to keep this at 0 and only optimize against the applied loads.

To make it easy to set up some rough boundary conditions the box control may help. This control is a representation of the bounding volume of your voxel model. Its orientation matches the 3d viewports camera orientation.

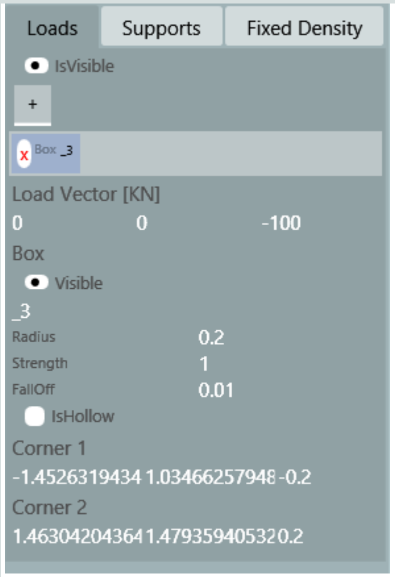


Clicking on any side edge or corner [grey elements] will toggle the boundary condition form totally fixed to totally free. This makes it easy for example to fix all the nodes on the ground with a single click.

In the example to the left, the left most wall of the box is set to “fixed” and you can see the small boxes in the 3d viewport that represent fixed nodes along the left side of the bounding volume.

The purple and blue elements will toggle applied loads. The rings apply point loads pointing downwards at the corners. The lines that come out of the sides apply area loads on the sides pointing along the corresponding side’s normal.

If you require more control over the distribution of supports and loads you can use the tabs that are located below the box control. These three tabs allow the definition of loads and supports for the current load-case [the density tab applies to all load-cases]


Loads | Supports | Fixed Density

● IsVisible

+

x Box _3

Load Vector [KN]

0 0 -100

Box

● Visible

_3

Radius 0.2

Strength 1

FallOff 0.01

IsHollow

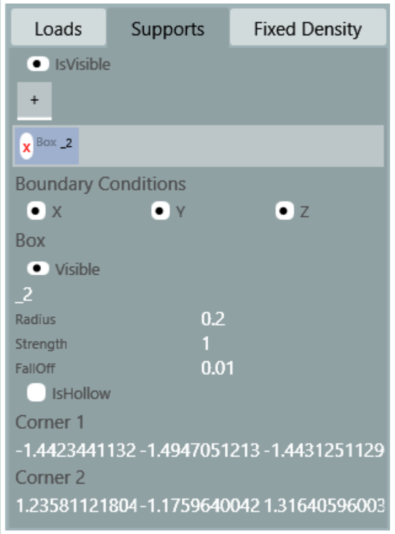
Corner 1

-1.4526319434 1.03466257948 -0.2

Corner 2

1.46304204364 1.47935940532 0.2

You can use the "+" button to add a load regions. These are the same type of geometric objects used in the geometry based field generator layer [points, lines , planes, boxes etc…]. You define the overall load in KN. This load will be divided across all the nodes that fall within the area of influence of the geometric object.


Loads | Supports | Fixed Density

● IsVisible

+

x Box _2

Boundary Conditions

● X ● Y ● Z

Box

● Visible

_2

Radius 0.2

Strength 1

FallOff 0.01

IsHollow

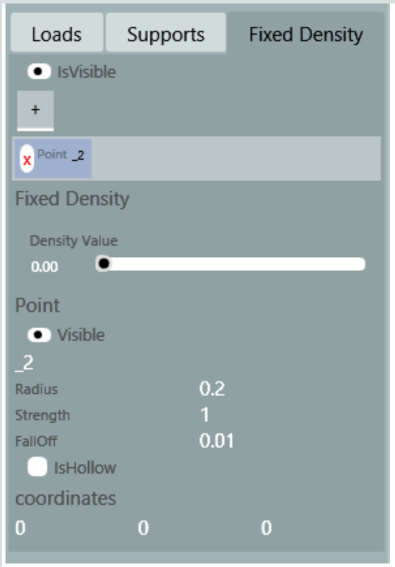
Corner 1

-1.4423441132 -1.4947051213 -1.4431251129

Corner 2

1.23581121804 -1.1759640042 1.31640596003

In the support tab use the "+" button to add geometric objects/ Using the XYZ checkboxes you determine the boundary conditions applied on all the nodes in the area of influence of selected geometric object.


Loads | Supports | Fixed Density

● IsVisible

+

x Point _2

Fixed Density

Density Value

0.00

Point

● Visible

_2

Radius 0.2

Strength 1

FallOff 0.01

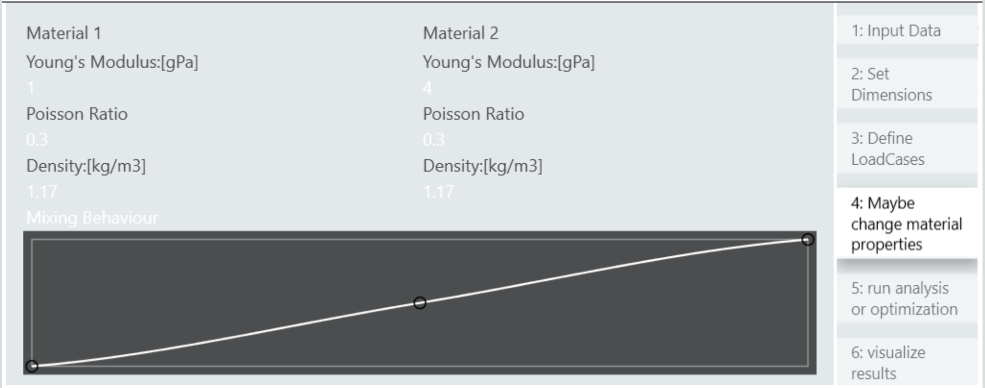
IsHollow

coordinates

0 0 0

Finally in the Fixed Density tab you can determine areas that have fixed density [from 0 to 1]. The voxels within these regions will maintain their density value during optimization. For example you can use regions with very small density to signify areas where no material should be distributed or use areas with a density of 1.0 to designate regions that have to maintain their material throughout the optimisation process.

3.3.4 Structures: step 4: Material definition



Material 1
Young's Modulus:[gPa]
1
Poisson Ratio
0.3
Density:[kg/m3]
1.17
Mixing Behaviour

Material 2
Young's Modulus:[gPa]
4
Poisson Ratio
0.3
Density:[kg/m3]
1.17

Here you can define the material properties of your model.

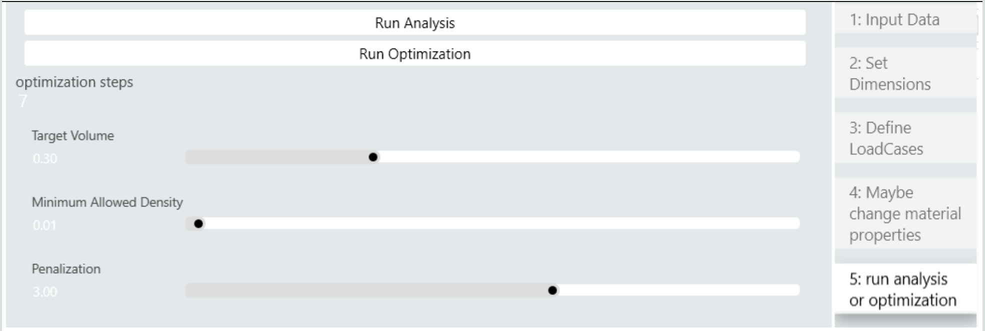
Currently monolith supports a dual material gradients

You define the mechanical properties of the two materials as well as the curve that will determine the ratio of their elastic moduli corresponding to a specific mixture of the two materials. In general if you mix 50% of a soft and a hard material you will not get a material with the exact mean of their elastic properties

In addition the micro structure of the material mixture will affect how the composite behave often in unpredictable ways

However as this information is difficult to come by and for an initial approximate analysis you can leave the mixing curve as is.

3.3.5 Structures: step 5: Run Analysis and Optimization



optimization steps
7
Target Volume
0.30
Minimum Allowed Density
0.01
Penalization
3.00

Here you can either run the analysis of the current model as is or attempt to redistribute the material so that a more lightweight structure is found

Monolith currently supports only linear elastic static analysis. This type of analysis is good when you want to assess the strength and stability of relatively rigid structures or objects but it does not work well for really flexible systems [e.g. very soft rubber

When you press the "run analysis" button, monolith will simulate the action of the forces in your current loadcase and determine how each voxel in your model deforms [this deformation is called the strain]. These minute deformations add up to determine the overall deformation of the object you analyse. In addition by knowing how much different regions in your model deformed monolith can determine the stresses acting locally at any point in space. These stresses will give you indication about which regions in your model might need reinforcement.

If you click on the "run optimization button", monolith will use a form finding method called "topology optimization" that will attempt to remove or redistribute the material in your model in order to find a structure that is maximally stiff for a given amount of material. This is also known as the "minimum compliance problem". Compliance is a measure of how flexible a structure is.

The optimization algorithm assumes that you can "magically" vary continuously the strength [let's say the porosity] of your material in space. It will iteratively shift material around and rerun the structural analysis until it finds a better solution. That also means that it can take quite some time depending on the resolution of your model.

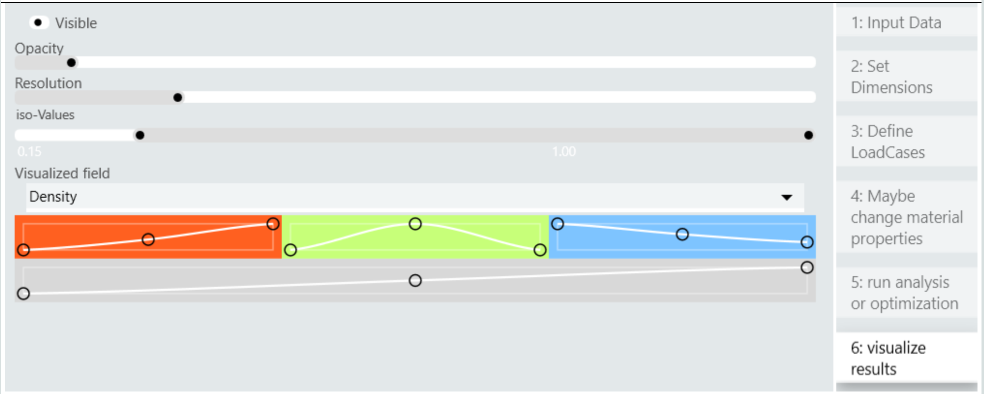
On top of your current structural model, the optimization module needs three more pieces of information

Target Volume: is the fraction of the overall volume of material left at the end of optimization. For example if you want to remove 50% of material from your box while maintaining as much strength as possible under your applied loads, you would set this value to 0.5

Minimum Material Density: Is the minimum fraction of material strength [maximum porosity] allowed for a given voxel. This can be set to a very small value [e.g. 0.001] which would mean that a voxel with this relative strength can be considered void. In some cases however you might want to simply determine a distribution between two materials of different strengths while not allowing any holes to emerge. In this case you can set this value to the ratio of the weaker material's elastic modulus to that of the stronger.

Penalization: this is an obscure settings that has to do with the specific algorithm used here during optimization. In general the highest this value is the more "high contrast" the final result will be, with sharp boundaries between high strength [and solid] and low strength [or void regions]. If you are interested in gradients rather than a solid/void condition set this value to around 1.0 otherwise leave it at 2.5 or 3.0.

3.3.6 Structures: step 6: Visualize Results



● Visible

Opacity

Resolution

iso-Values

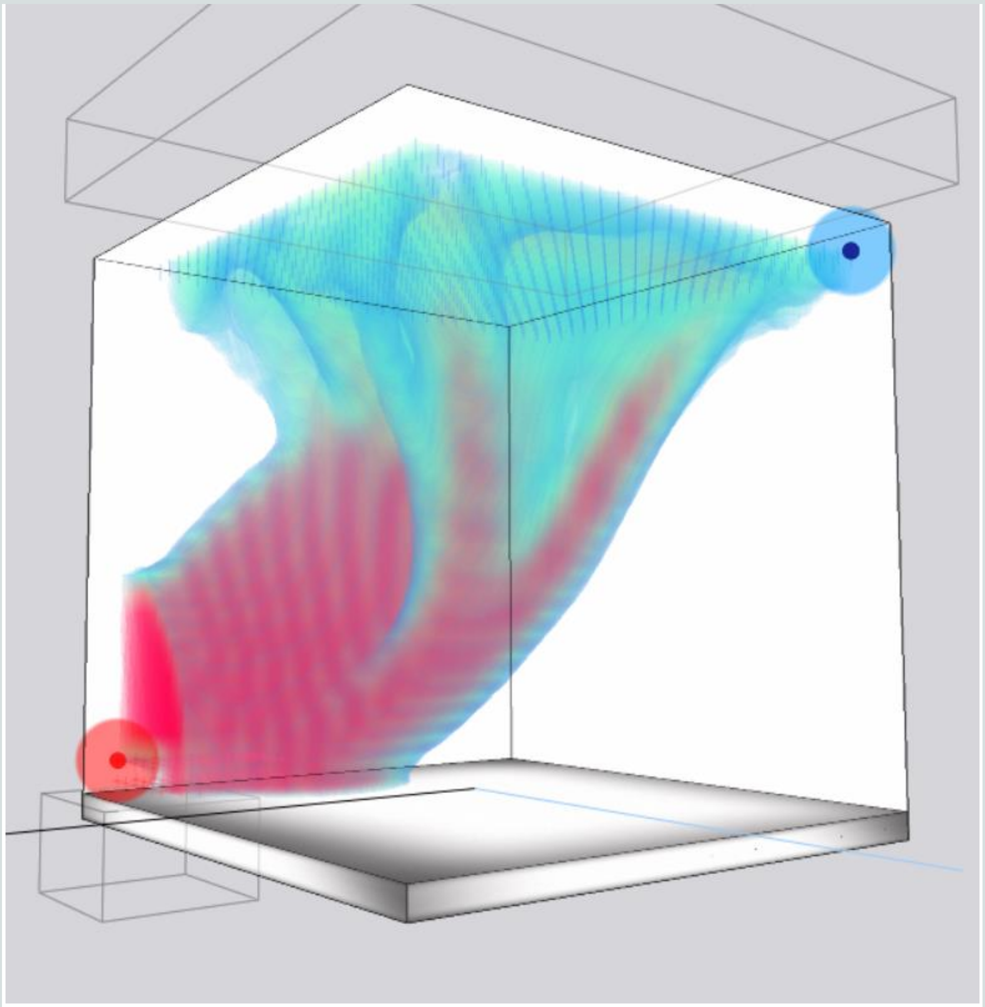
0.15

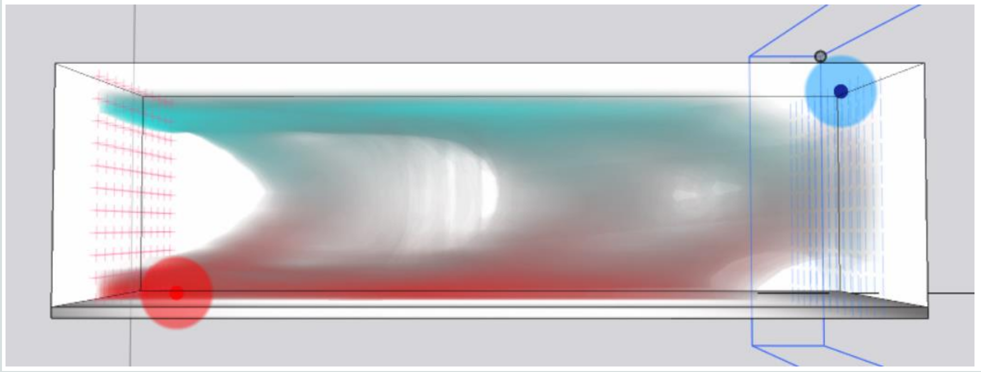
1.00

Visualized field

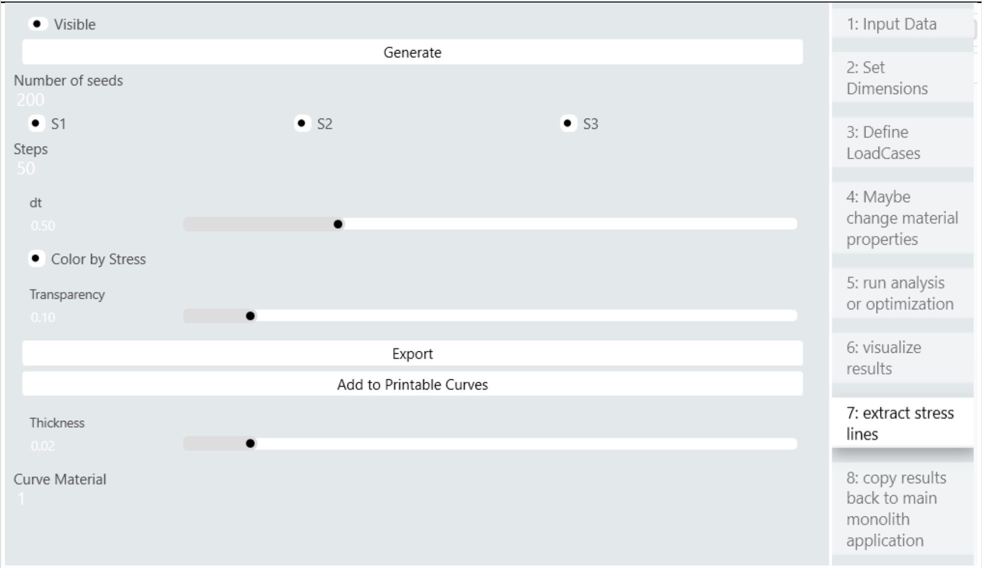
Density

The purpose of this panel is to help you inspect the analysis model. You can select between various visualization modes as well as define the colour gradient and clipping iso values for the density field. . The 4 interpolators below the combo box determine the colour gradient [transfer function] used. In the example above we have chosen a stress visualization. The 4 interpolators define a gradient that will be blue for 0 stress and red near the maximum stress. Also low stress regions will have a very low alpha value [will be transparent].





3.3.7 Structures: step 7: Extraction of stress lines



● Visible

Generate

Number of seeds
200

● S1 ● S2 ● S3

Steps
50

dt
0.50

● Color by Stress

Transparency
0.10

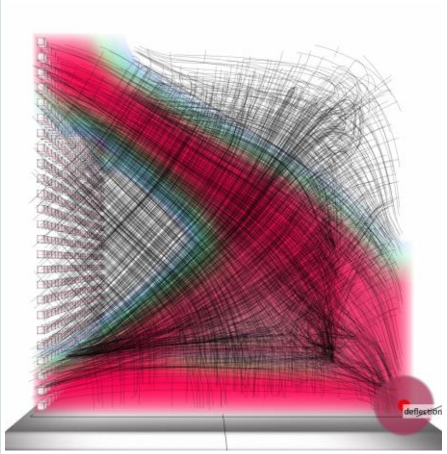
Export

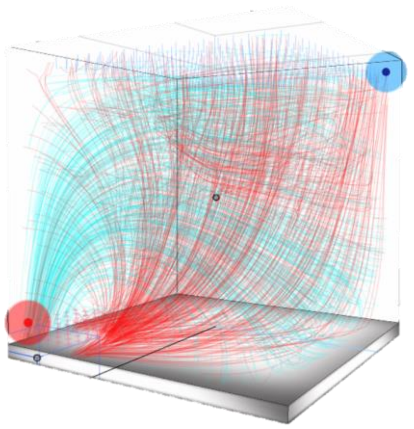
Add to Printable Curves

Thickness
0.02

Curve Material
1

1: Input Data
2: Set Dimensions
3: Define LoadCases
4: Maybe change material properties
5: run analysis or optimization
6: visualize results
7: extract stress lines
8: copy results back to main monolith application





Using this panel you can extract the principal stress lines from the analysed model. These lines are useful in guiding fibre placement or other reinforcement types within the bulk of the material. If you want you can add them to the printable curves set of monolith and they will be rasterized along with the rest of the model. As you can see from the photos on the right you can create very fine structures without having to actually convert the lines to meshes.

The following controls are defined in this panel:

Visible: use to turn on and off visibility of stress lines

Number of seeds: determines the number of stress lines drawn. These will be randomly seeded throughout the analysis domain.

S1/S2/S3: turn on these checkboxes to determine which of the three principal stress directions are used. S1 is the minimum eigenvalue direction [and usually will follow the compression paths]. S3 is the maximum direction [and will usually reveal tension paths]. S2 is the middle eigenvalue of the stress tensor.

Steps: The number of integration steps along each stress line

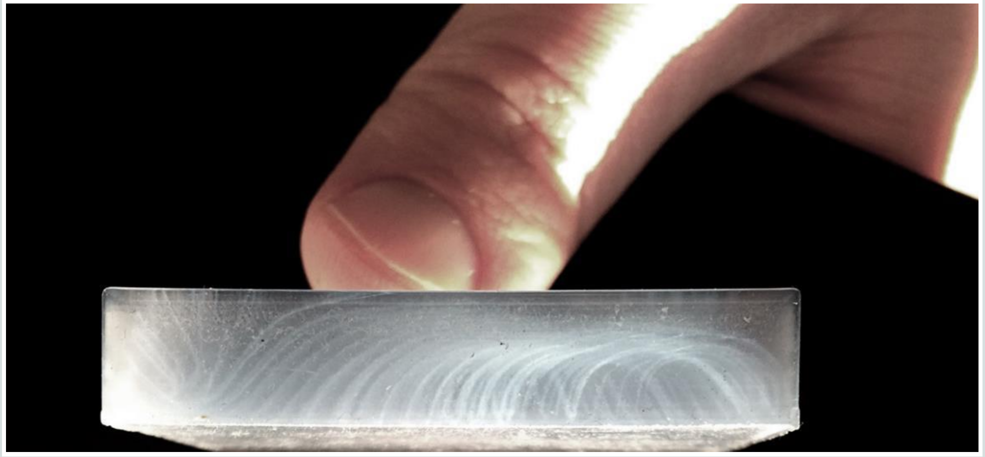
Dt: The integration step as a fraction of the voxel size.

Colour By stress: When on stress lines will be coloured [red or blue] depending on whether the corresponding voxel is in compression or tension along the stressline.

Transparency: the transparency of the stress lines

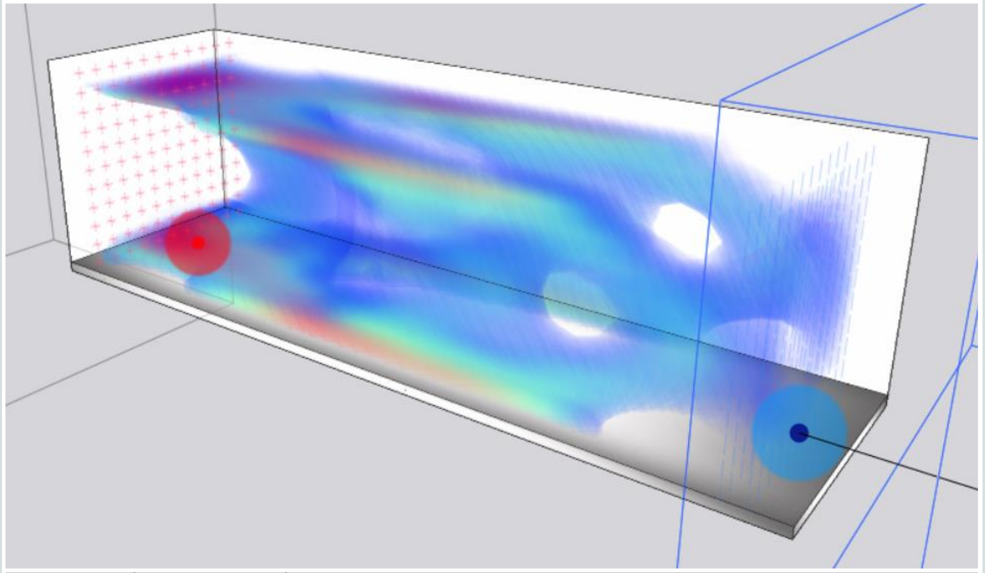
Export: use this option to export stress lines as curves to a 3dm file.

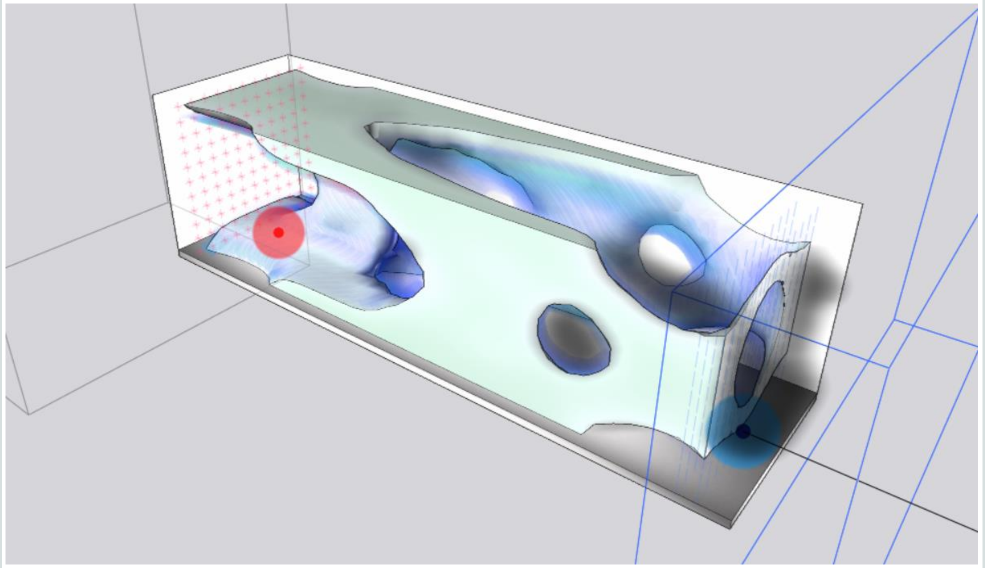
Add Printable Curves: Use this option to transfer the stress lines to the printable curves group. The selected thickness and material [chosen below] will be applied to these curves and they will be composited during rasterization.



3.3.8 Structures: step 8: Output

Finally you can output the optimization results back to the main monolith compositing system. When you click the "output" button a new layer will be created in monolith whose shape channel and material gradient channels are set to be the same as in the optimized model.





4 I/O

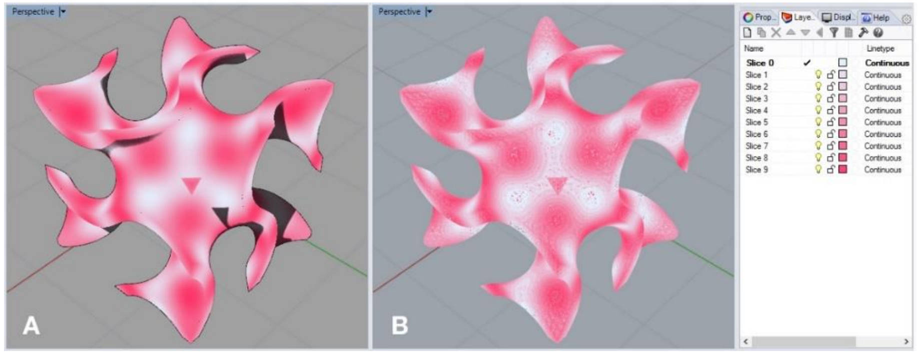
4.1 Import/Export voxel files

Monolith has its own file formats for saving voxel data. The two formats are “lith” and “vol”. The lith” file stores the monolith layer stack with all the parameters and settings needed to recreate the model. In addition your texture mapping settings and structural analysis models if any are stored there as well. The vol file is a lightweight format that contains only the voxel values of the current model. You might want to export a model as a vol file if you want to for example use it as a rasterization pattern in another model.

In additions you can import a stack of images from a designated folder that will automatically be converted to a freehand Layer in monolith.

4.2 Export Meshes

Monolith can export mesh files in various formats including stereolithography (.stl), VRML 2.0 (.wrl), (.obj) and Rhino (.3dm) files. Mesh files can also be saved out as slices where each individual slice represents a Boolean intersection between the Shape and Material channels. If saving to a Rhino multimaterial slice file, then the slices will be saved to individual layers for easier post-processing. Stereolithography multimaterial slice files will save each mesh slice as a new .stl file.



[A] VRML mesh exporting supports per vertex colouring. [B] Rhino multimaterial slice files will save individual slices on separate layers for easier post-processing.