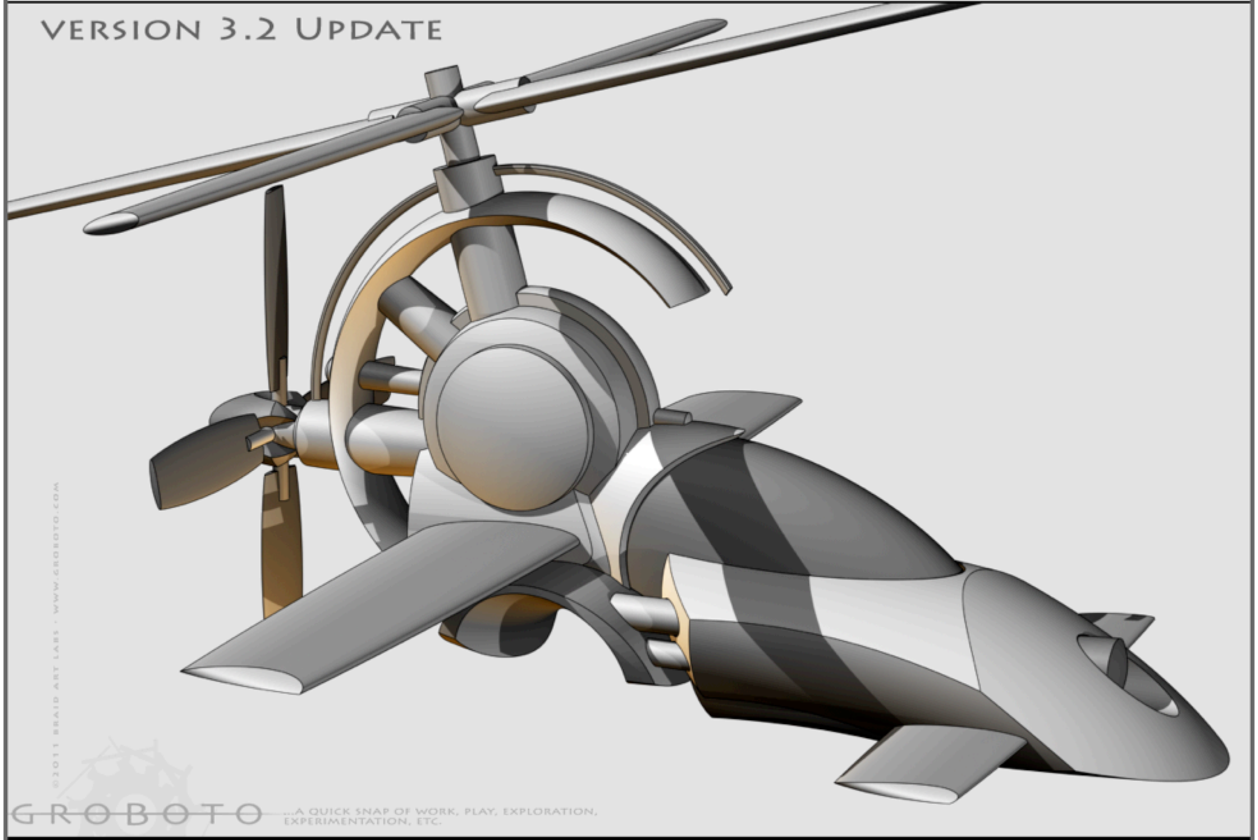
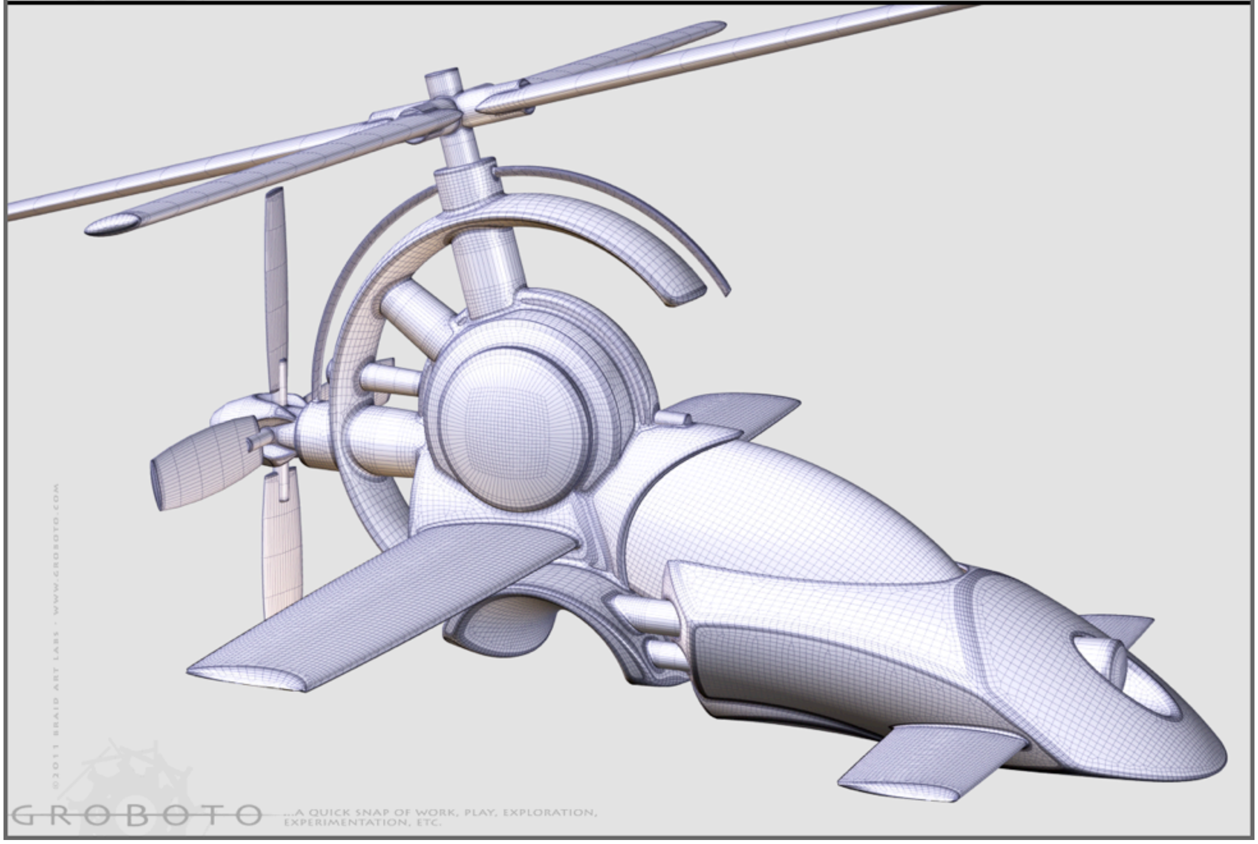
GroBoto Mesh & Map Notes

version 3.2 Update

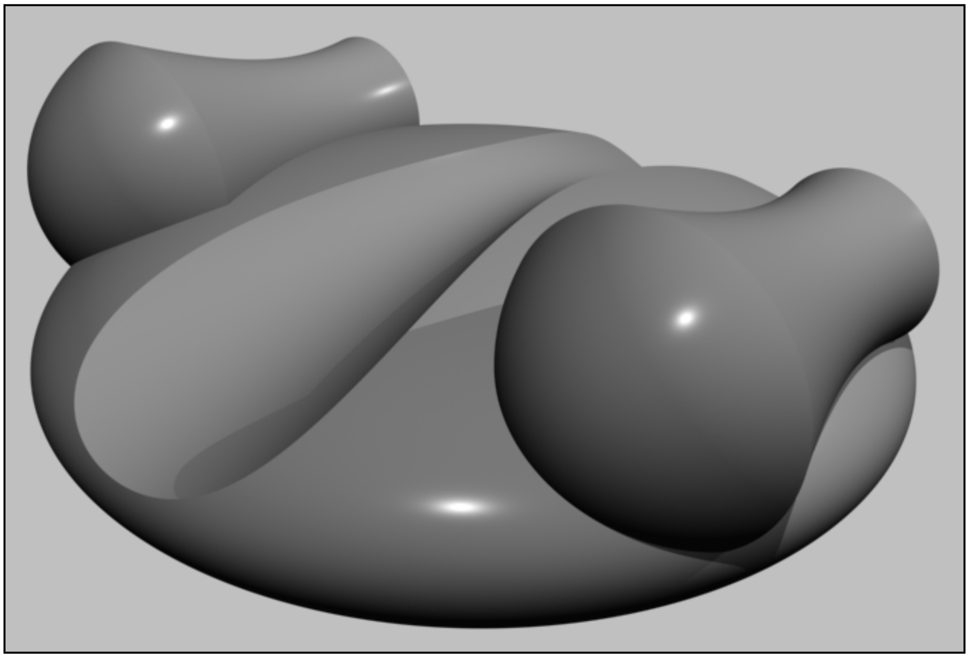

©2011 Braid Art Labs · www.groboto.com
GROBOTO
...a quick snap of work, play, exploration, experimentation, etc.


©2011 Braid Art Labs · www.groboto.com
GROBOTO
...a quick snap of work, play, exploration, experimentation, etc.

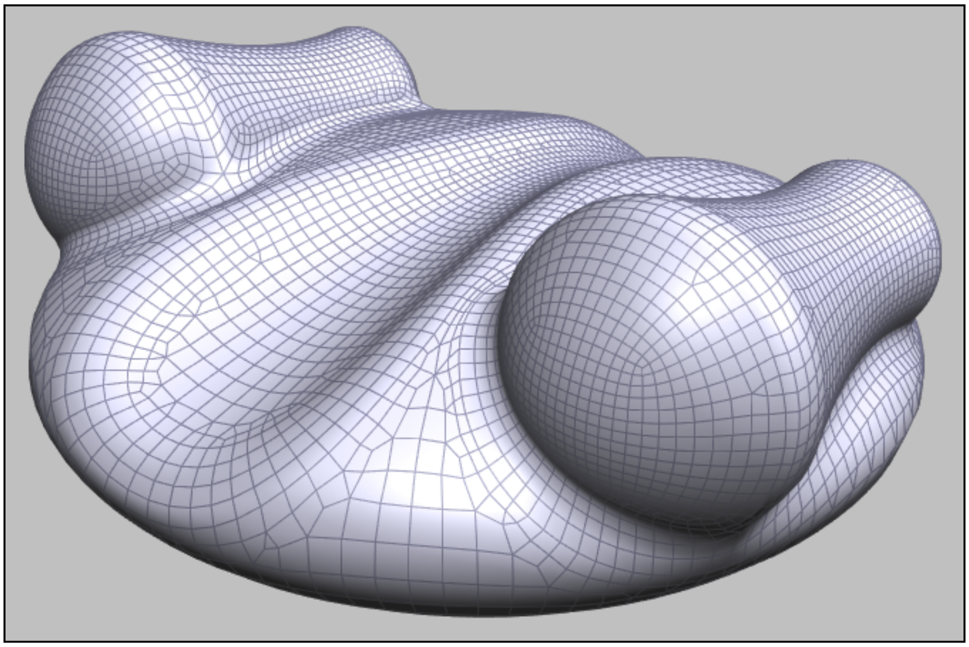
Contents

Intro

The bridge that connects Groboto's object-based models with the wide world of polygonal modeling and rendering is Unified Mesh Output. Any Groboto scene consisting of any number of objects and boolean clusters:



can be automatically converted into a single continuous polygonal mesh:

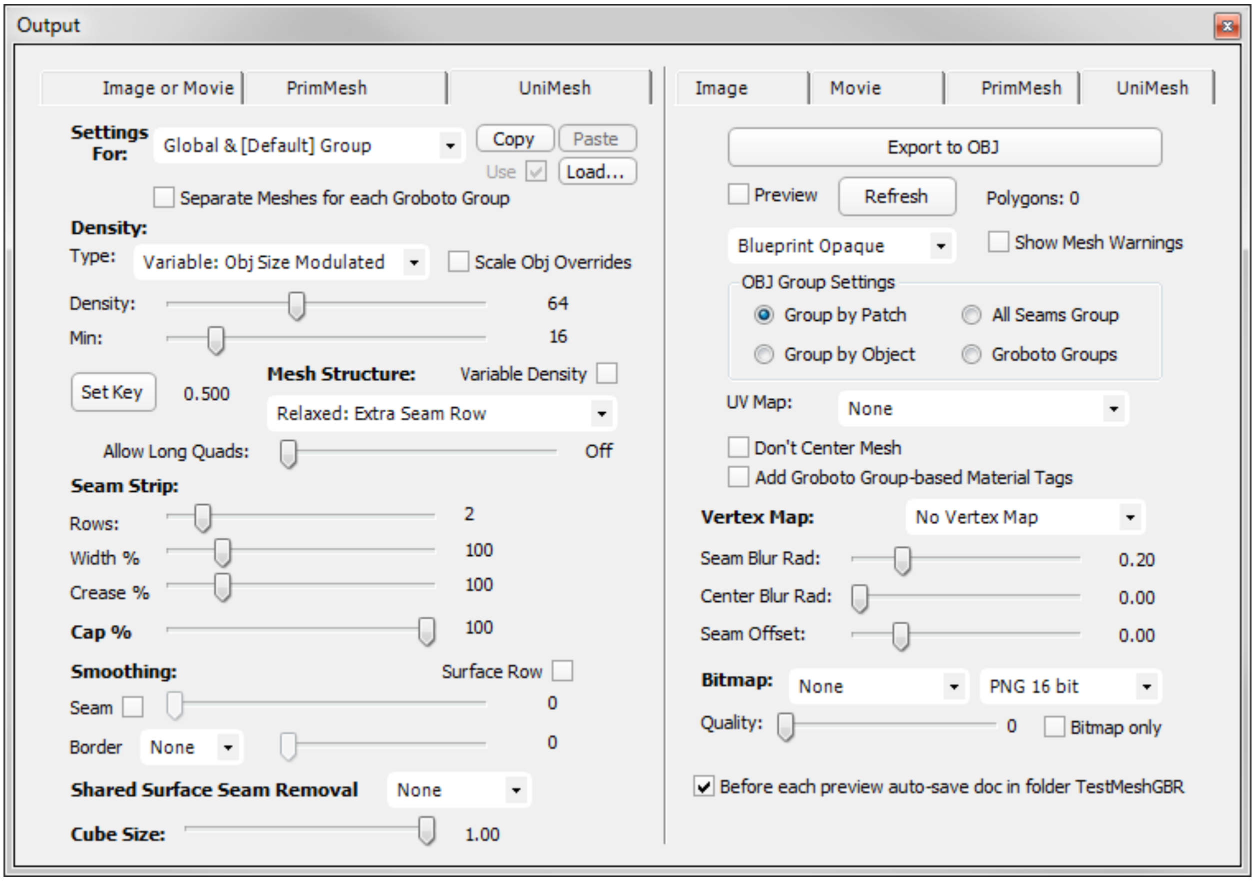


This automated Mesh Generation — with options for recasting simple primitive forms into sophisticated, stylized surfaces — is the key to using Groboto as a modeling tool. Groboto allows you to build shapes of arbitrary complexity by combining simple primitive shapes, but this by itself gives you only a schematic representation of the model, where the primitive shapes are separated by sharp visible seams. To go any further, to create a more organic interaction between the component shapes, the model has to be converted into a single continuous mesh.

With more complex Groboto models you will frequently want to convert the model into several disconnected mesh pieces rather than a single one. This is handled by organizing primitive objects into Groboto groups and creating a separate continuous mesh for each group (see page 68). By default Groboto creates single unified mesh from the entire scene.

Output Panel

The UniMesh Output Panel controls Unified Mesh conversion (File->Export).



Settings For: Global & [Default] Group

Use ☑

☐ Separate Meshes for each Groboto Group

Density:

Type: Variable: Obj Size Modulated

☐ Scale Obj Overrides

Density: 64

Min: 16

0.500

Mesh Structure: Variable Density ☐

Relaxed: Extra Seam Row

Allow Long Quads: Off

Seam Strip:

Rows: 2

Width % 100

Crease % 100

Cap % 100

Smoothing: Surface Row ☐

Seam ☐ 0

Border None 0

Shared Surface Seam Removal None

Cube Size: 1.00

☐ Preview

Polygons: 0

Blueprint Opaque

☐ Show Mesh Warnings

OBJ Group Settings

● Group by Patch

○ All Seams Group

○ Group by Object

○ Groboto Groups

UV Map: None

☐ Don't Center Mesh

☐ Add Groboto Group-based Material Tags

Vertex Map: No Vertex Map

Seam Blur Rad: 0.20

Center Blur Rad: 0.00

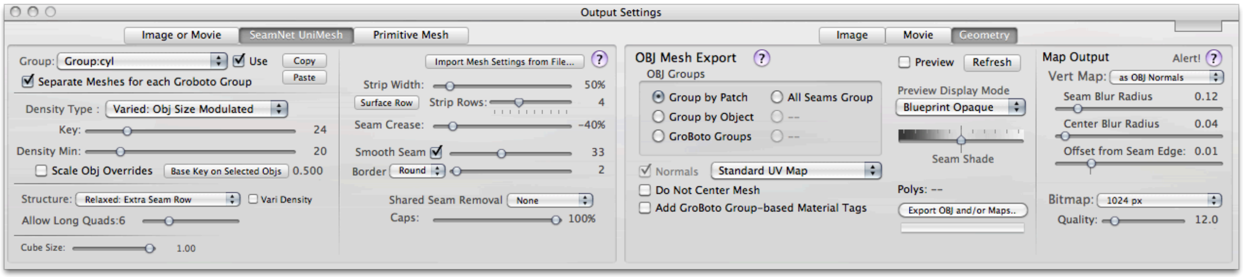
Seam Offset: 0.00

Bitmap: None PNG 16 bit

Quality: 0 ☐ Bitmap only

☑ Before each preview auto-save doc in folder TestMeshGBR

Above:Windows   Below:Mac OSX (zoom in for a better look).



Group: Group:cyl ☑ Use

☑ Separate Meshes for each Groboto Group

Density Type : Varied: Obj Size Modulated

Key: 24

Density Min: 20

☐ Scale Obj Overrides 0.500

Structure: Relaxed: Extra Seam Row ☐ Vari Density

Allow Long Quads:6

Cube Size: 1.00

Strip Width: 50%

Strip Rows: 4

Seam Crease: -40%

Smooth Seam ☑ 33

Border Round 2

Shared Seam Removal None

Caps: 100%

OBJ Mesh Export

OBJ Groups

● Group by Patch

○ All Seams Group

○ Group by Object

○ --

○ GroBoto Groups

○ --

☑ Normals Standard UV Map

☐ Do Not Center Mesh

☐ Add GroBoto Group-based Material Tags

☐ Preview

Preview Display Mode: Blueprint Opaque

Polys: --

Map Output

Vert Map: as OBJ Normals

Seam Blur Radius 0.12

Center Blur Radius 0.04

Offset from Seam Edge: 0.01

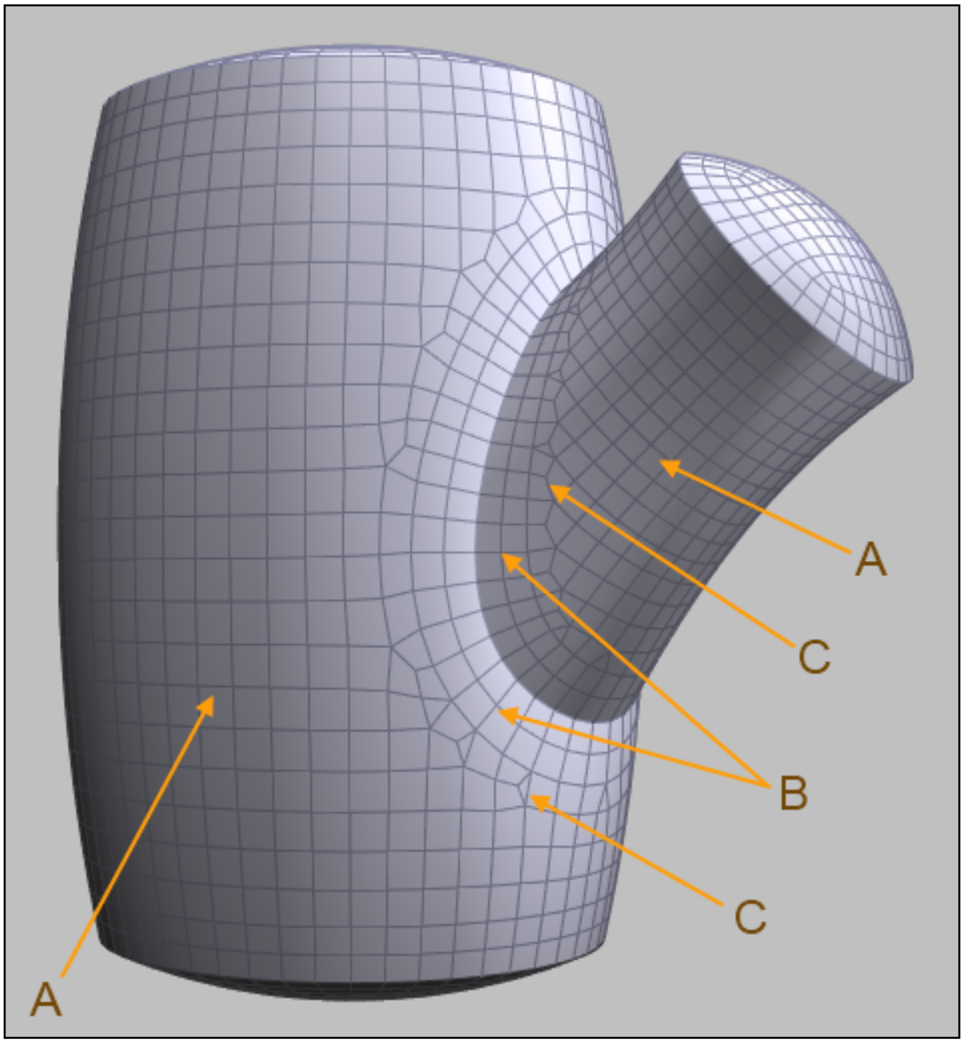
Bitmap: 1024 px

Quality: 12.0

To better understand the meaning of various mesh control parameters you might want to open a simple Groboto scene and generate mesh repeatedly by clicking Refresh in the Output dialog, as you enable various options and change parameters.

Note: Most of the interface illos in this doc are from the Windows version. The controls are identical, although the arrangement and exact titles may vary. See GroBoto Online Docs (Mac or Win Interface sections), for item-by-item reference.

Looking at a typical Groboto unified mesh, we can clearly see several different areas of the mesh:



A. Native mesh of each primitive. This mesh consists of orderly rows and columns of quads covering each of the simple primitive shapes.

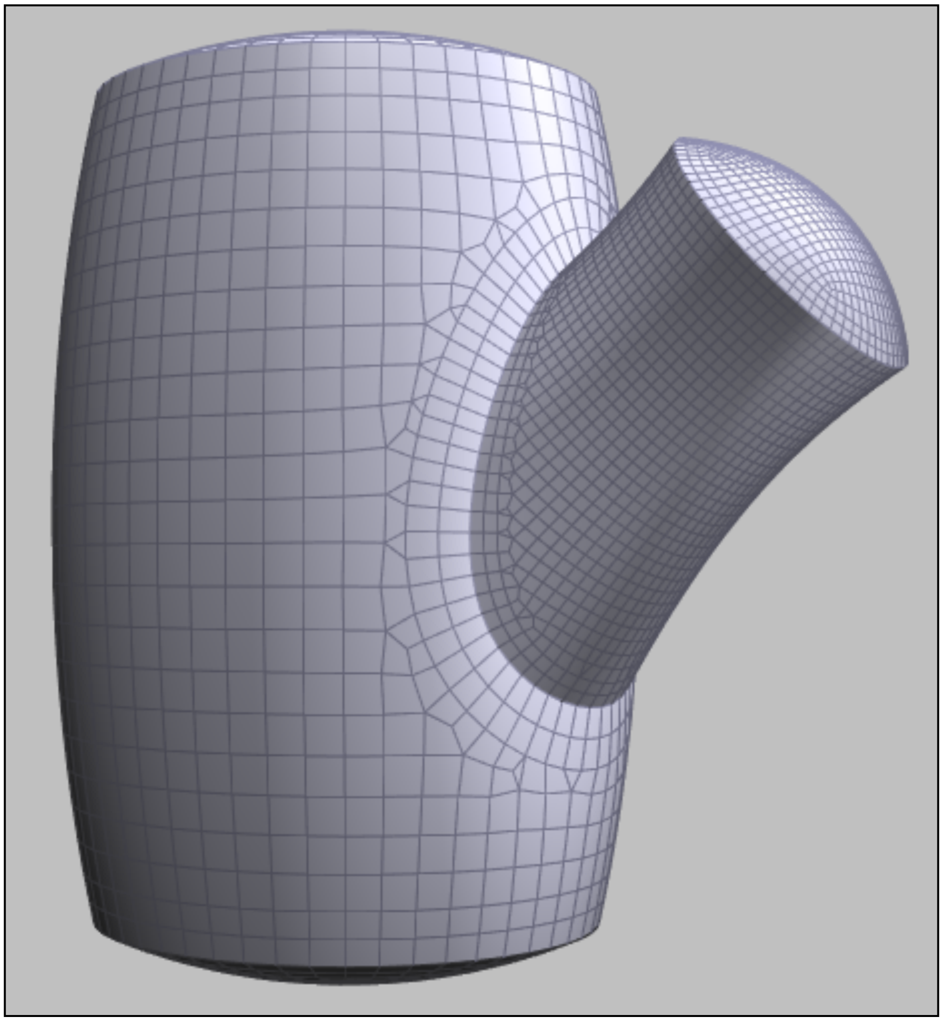
B. Seam strips. Where primitive shapes collide with each other, on both sides of the intersection curve we have orderly rows of quads following the intersection curve (in this case two rows on each side).

C. Most of the time native mesh and seam strips have different orientation, thus it becomes necessary to sew them together to create a continuous mesh, by inserting a somewhat disorderly sequence of quads and triangles between them.

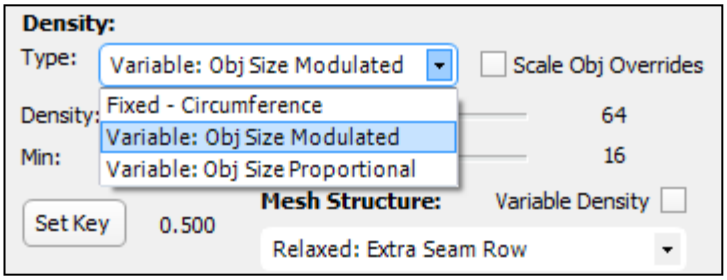
Mesh Density

The most important mesh property is polygonal density; this determines the number of quads in the mesh and their size. What you directly control in Groboto is density of the native mesh (A). This could be set independently for each object in the scene. The density in the areas B and C is automatically derived from the density of nearby areas A to minimize the density discrepancy and provide the smoothest possible transition between different areas.

Density management is the key to creating good unified mesh in Groboto. Generally what you want is to have similar quad sizes on any two objects that intersect each other. The quad size of the seam strip between them is always set to the average value of the object quad sizes, thus all three quad sizes will be close. Groboto will always generate continuous mesh, even if quad sizes on neighboring objects are very different, but the mesh will have more irregular quads and triangles:

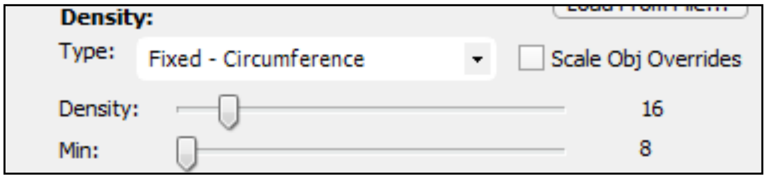


Density could be set individually for each object, but in most cases you resort to individual density setting only for some specific objects in the scene, while for most other objects relying on automatic density setting. Groboto has three automatic types of density setting, chosen in the Density Type pop-up:



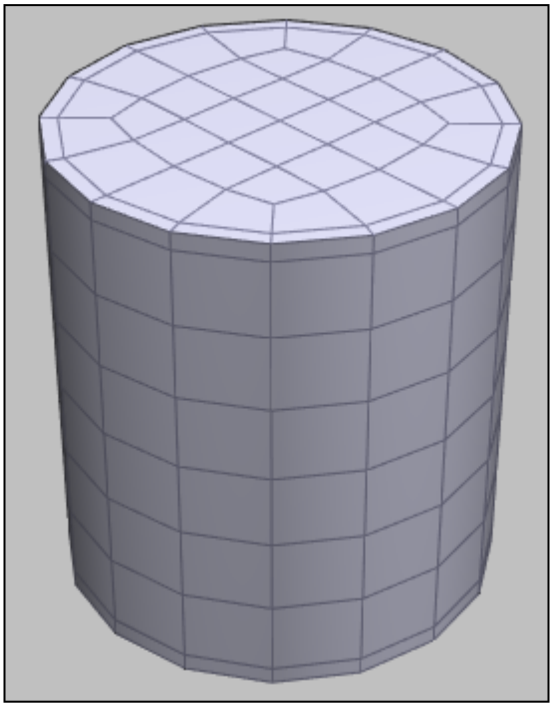
Density:
Type: Variable: Obj Size Modulated
☐ Scale Obj Overrides
Density: 64
Min: 16
Set Key 0.500
Mesh Structure: Relaxed: Extra Seam Row
Variable Density ☐

1. Fixed - Circumference. This means every object, large or small, gets the same number of quads around its circumference. The number of quads is specified by the Density slider.



Density:
Type: Fixed - Circumference
☐ Scale Obj Overrides
Density: 16
Min: 8

This means larger objects will be getting larger quads, quad size being proportional to object size. For instance, if Density is set to 16, we get 16 subdivisions around the cylinder. The quads are square, so how many subdivisions we get along the axis of the cylinder depends on its length:

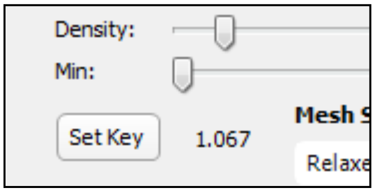


2. Variable: Obj Size Modulated. The larger the object, the more quads around its circumference, but the number of quads grows slower than the size, so larger objects still get larger quads.

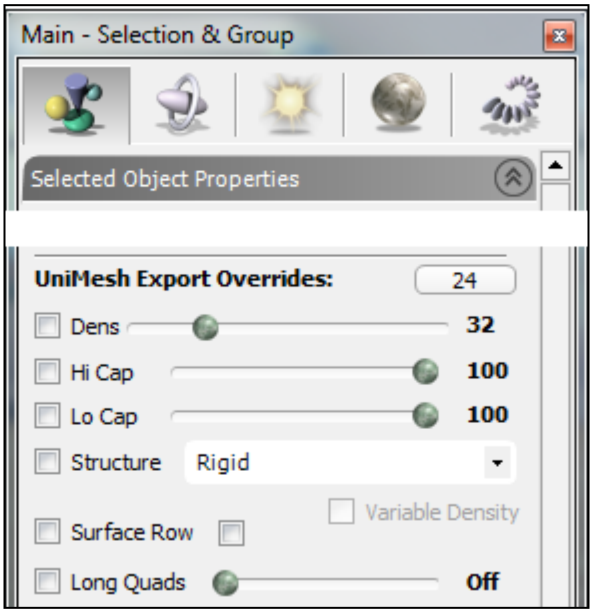
3. Variable: Obj Size Proportional. The number of quads grows proportional to the object size, thus quad *size* is the same for large and small objects.

The second type is a compromise between the first and the third type, producing density somewhere in-between. With the second and third types the density (number of quads around the circumference) is different for different objects; to have greater control of this variable density you can also use the Min slider to set the minimum density. No matter how small, any object will have at least the number of divisions around its circumference indicated by the Min slider setting.

When using the second and third type you might want to give exact meaning to the value indicated by the Density slider. With these types the number of divisions around each object depends on the size of the object. For what object size would you get the number of divisions indicated by the slider? By default this is set to a fairly typical Groboto object size 0.5. This might or might not be related to object sizes used in your scene. If you select any object in your scene and click the Set Key button the number to the right of the button will be set to the size of the selected object and that size will become the key size: the selected object will get the number of divisions indicated by the Density slider; smaller objects will get fewer divisions, larger objects more divisions.



If you find that for some objects the automatic density setting is not right, you can override it with a manual setting. Like all object properties this is controlled from the Selected Object Properties (leftmost) tab of the Main Groboto panel (on the right side of workspace in the default layout).

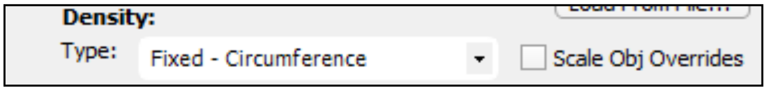


UniMesh Export Overrides: 24
☐ Dens 32
☐ Hi Cap 100
☐ Lo Cap 100
☐ Structure Rigid
☐ Variable Density
☐ Surface Row ☐
☐ Long Quads Off

The UniMesh Export Overrides section there has the Dens checkbox and its corresponding slider. There is also a button above this slider with a number in it. This number shows the current automatic density setting for the selected object - the number of divisions around it when the override Dens checkbox is **not** turned on. You can click on the button to set the slider to the same value. Now you can move the slider to increase or decrease this value as desired, and click the checkbox to put the manually set value into effect (it will be used the next time you Preview or Output unified mesh). If you uncheck the Dens checkbox the density for the selected object will revert to the automatic value shown in the button.

If you select more than one object you can enable the manual setting and set the manual density to the same value for all the selected objects. If the checkbox appears in the undefined state it means the manual setting is currently enabled for some objects in the selection and disabled for others. If the slider value appears grayed-out it means the manual density setting is currently different for different objects (the number shown is the average for selected objects).

When you enable manual density overrides for some objects, their density is naturally no longer affected when you change the automatic density in the Output dialog box. Sometimes, however, this is exactly what you want - to increase or decrease the density for all objects, with or without manual density overrides. To do this, check the Scale Obj Overrides checkbox and adjust the Density slider in the Output dialog box as desired.

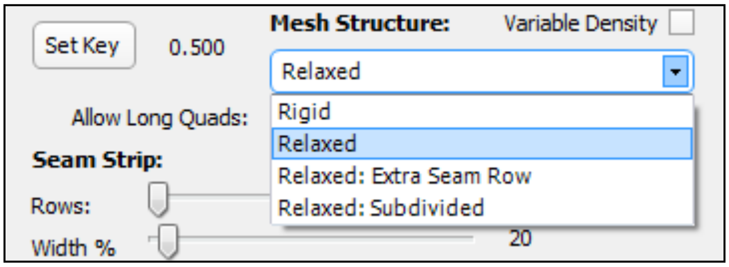


Density:
Type: Fixed - Circumference
☐ Scale Obj Overrides

This will change the density of all objects proportionally (e.g., double it). If you uncheck this checkbox, the manual density settings will not be affected by the *subsequent* movements of the Density slider in the Output dialog, but they will retain whatever changes occurred to them while the checkbox was checked.

Mesh Structure

The other important characteristic that affects the entire mesh is set in the Mesh Structure pop-up in the Output dialog box.



There are four choices:

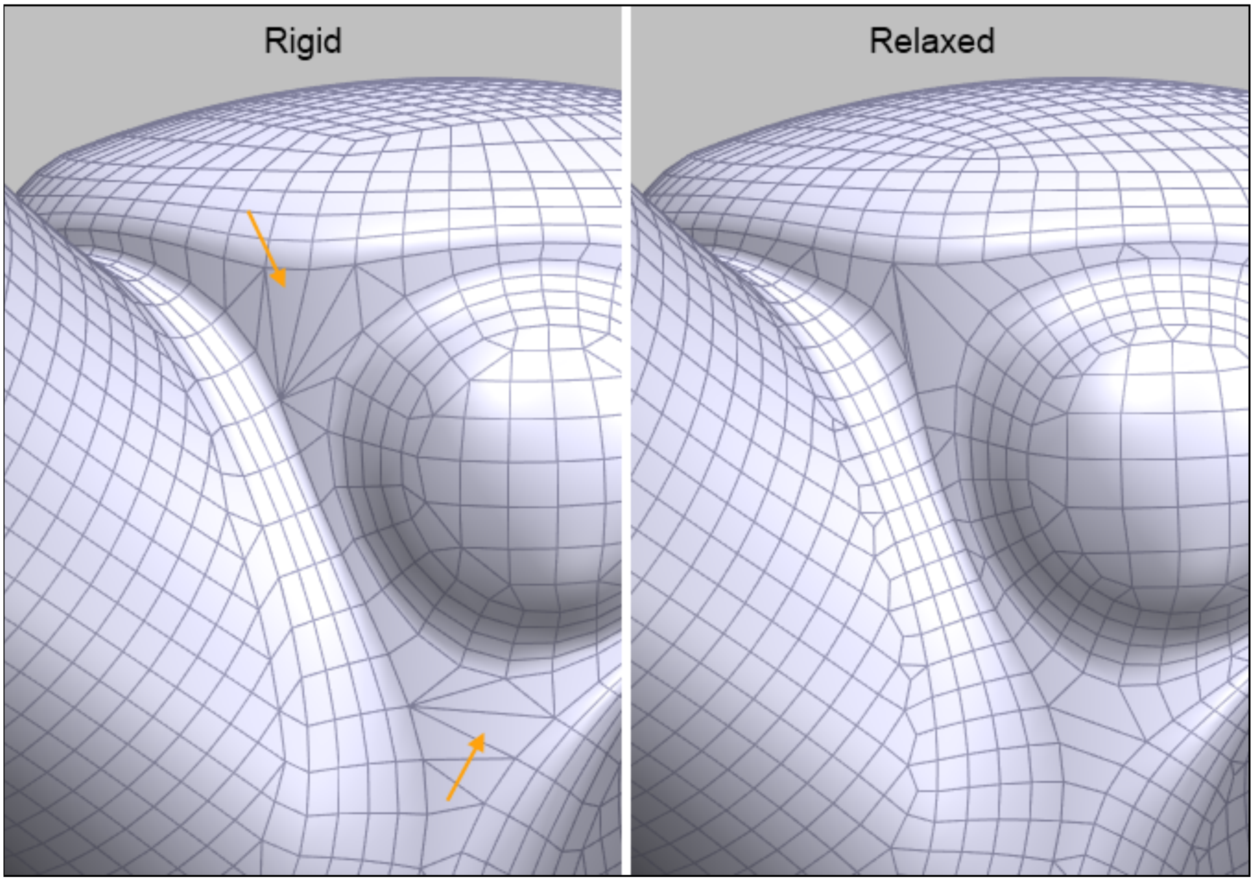
1. Rigid. In this mode vertices of the native mesh (A) have predefined positions not affected by interaction with other objects. This is typically the least attractive of all mesh structures - because the vertex positions cannot be modified, the quads and triangles in the transition area C are all different shapes and sizes, making it look pretty disorderly. The advantage of this mode is its robustness - no matter how complicated the arrangement of objects, this mesh can never overlap itself.

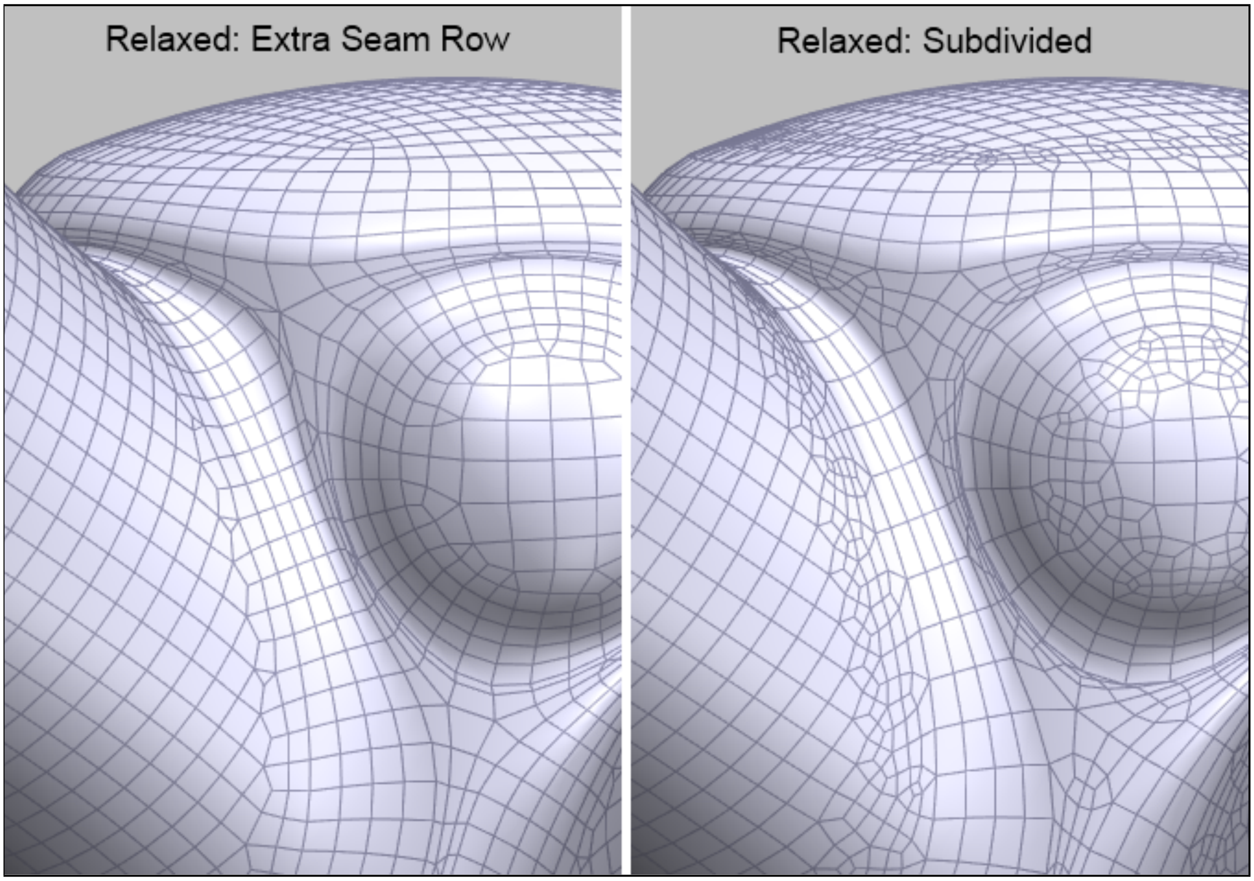
2. Relaxed. In this mode the native mesh is allowed to stretch and compress when interacting with seam strips, which smoothes out the disorderly area between the native mesh and seam strips. Typically this results in a much nicer mesh than Rigid. However, in some cases when the native mesh is squeezed in a narrow channel between two seam strips, and the density is low enough, the vertices of the native mesh might shift too much and slide into the seam strip, creating an overlapping mesh.

3. Relaxed: Extra Seam Row. Similar to Relaxed; in addition to allowing existing native mesh vertices to stretch and relax, it inserts one extra row of vertices into the relaxed area. This is very helpful in tight spots and narrow crevices between the seam strips, where relaxation of existing vertices by itself does not create a nice mesh simply because there are not enough vertices for that. This is the default unified mesh structure, and usually the best when Seam Smoothing is on, as it typically is (see below).

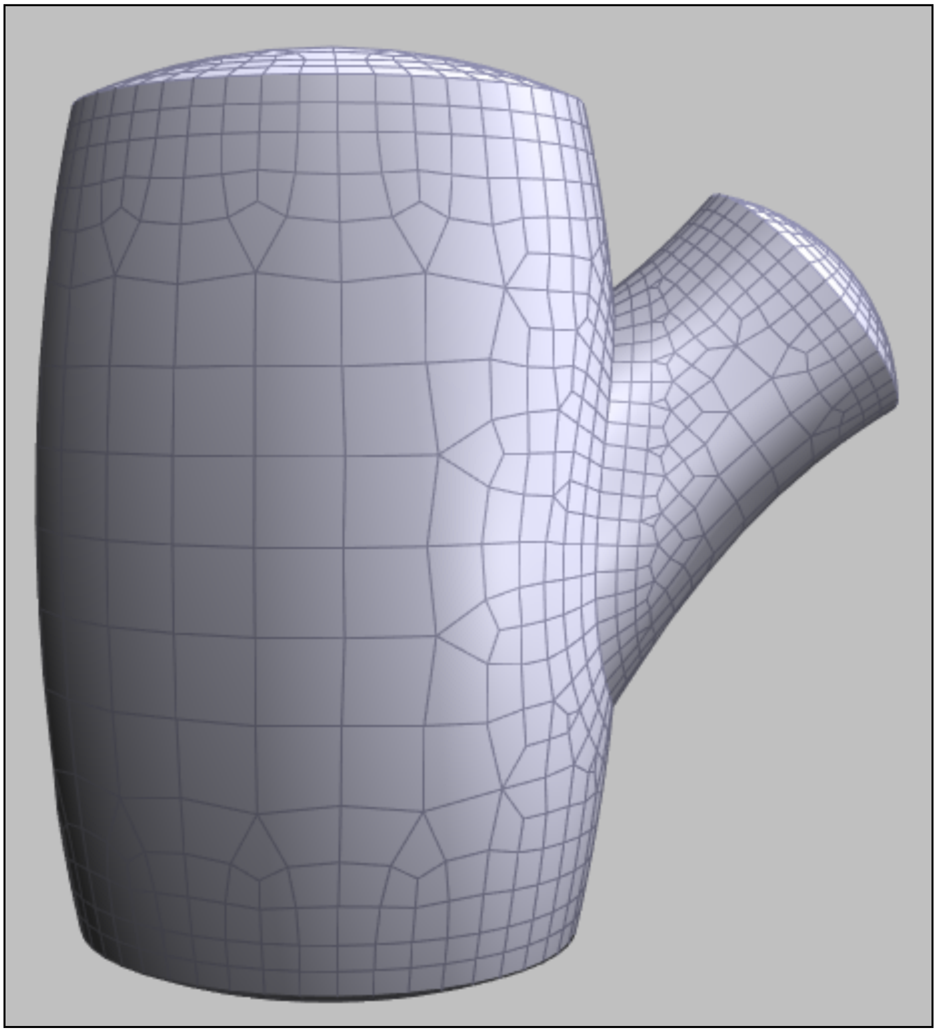
4. Relaxed: Subdivided. In this mode all quads and triangles in the disorderly area C are subdivided one level, creating a mesh that has only quads (the only mesh mode guaranteed not to have any triangles, although in other modes the number of triangles is always small). This mode produces considerably greater number of polygons than other modes (but the increase is far less than by a factor of 4, because subdivision occurs only in the area C and nearby, not affecting most of the native mesh).

Images below show one example of applying these four mesh structure settings. Orange arrows point to areas which, at this fairly low polygonal density, are problematic. Higher density cures problems of this sort no matter what mesh structure you use, but typically you'll be trying to avoid the expense of higher density.



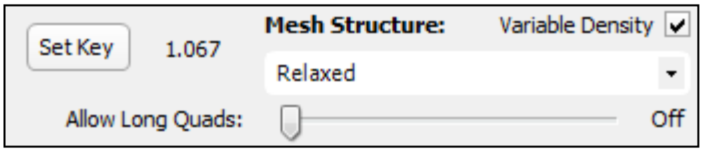


With any of these choices except Rigid you can also check the Variable Density checkbox. This creates larger quads on the parts of the surface farther away from the seams, by combining 2x2 sets of original density quads into larger quads.



This can reduce the total polygon count considerably, especially when using higher density. These larger quads are never generated near intersection seams, so they do not create any mesh quality problems in tight spaces between the seams and near corners. However, if you plan to further manipulate the mesh after exporting it from Groboto, reducing mesh density on some parts of the surface with this option might reveal its polygonated nature when using certain types of tools later.

There is another option that can significantly reduce the number of quads in the Groboto unified mesh, controlled by the slider called Allow Long Quads.



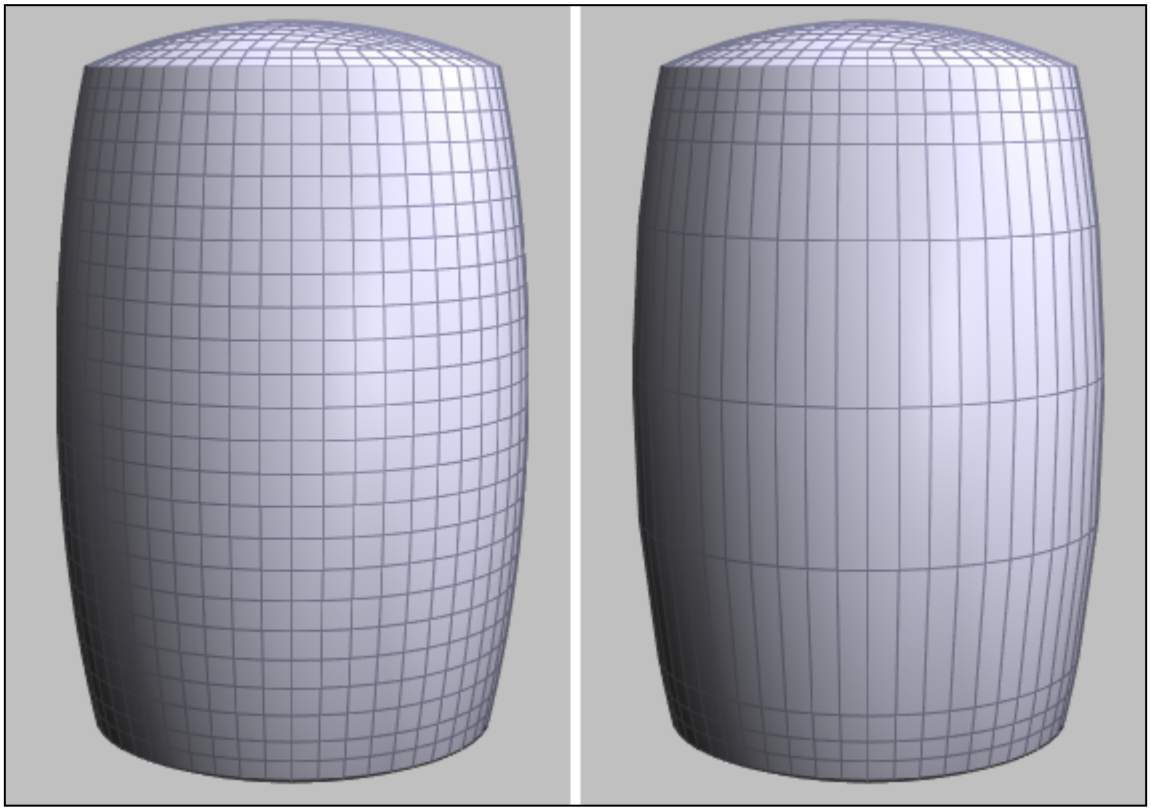
1.067

Mesh Structure: Relaxed

Variable Density ☑

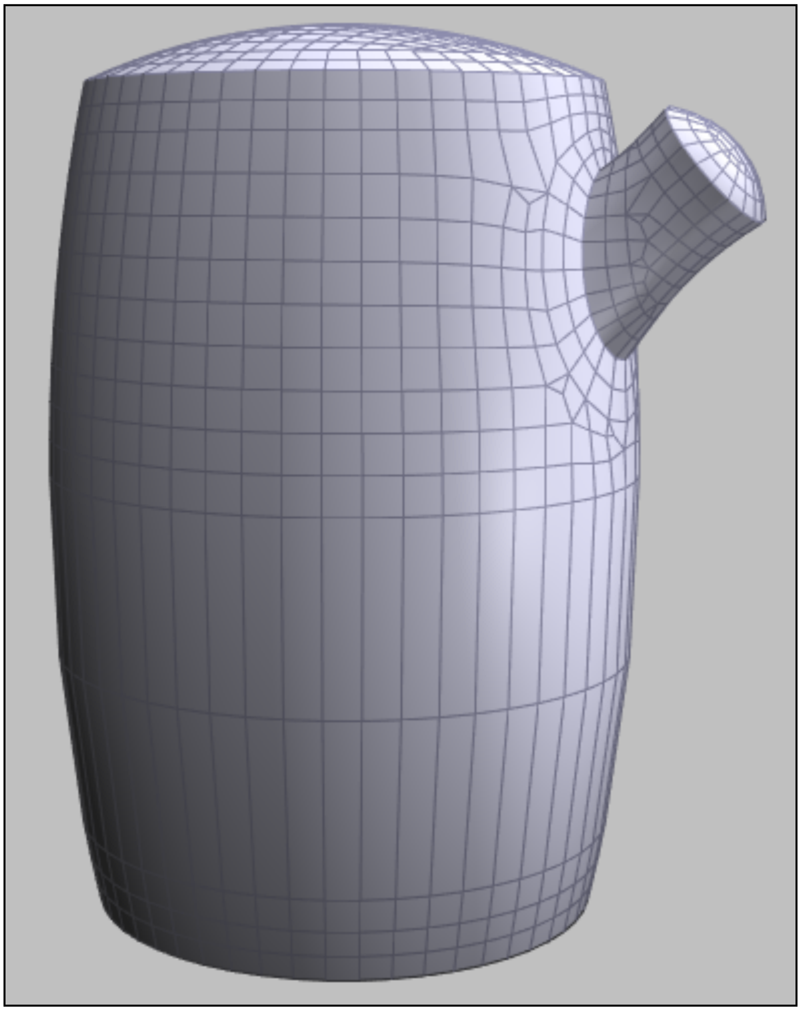
Allow Long Quads: Off

It is off by default; the image below right illustrates its effect on the mesh (Long Quads = 5):



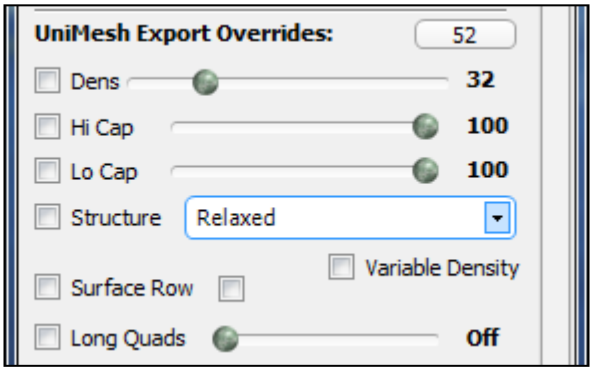
This option can be used on any objects whose native mesh contains uninterrupted rows of quads going all the way around the circumference - certainly cylinders and cones, but also spheres, ellipsoids and other 'round' objects. In a mesh of this type quads could be merged in the axial direction, producing elongated quads and reducing overall polygon count. On cylinders and cones doing this is always harmless, since these surfaces are not curved in the axial direction. On other surfaces it might be OK too, if the curvature in the axial direction is low enough. Elongated zeppelin ellipsoids and long hyperrods and hypertubes are good examples. The slider specifies the maximum number of quads allowed to merge into a single elongated quad. This is important when applying it to objects that are somewhat curved in the axial direction (as opposed to cylinders and cones, which are completely straight). Using low settings like 2 or 3 might be completely harmless, while a much higher setting could produce obvious polygonation. As with other options that reduce the number of quads, also take into account what kind of mesh modification you might want to do after exporting the model from Groboto. On a cylinder having elongated quads of any length is harmless, but not if you plan to bend it later.

When applying this feature you don't need to worry about its effect on intersections between objects. Intersections would be severely mangled if quads near them were elongated, but Groboto will never do that - long quads are only produced when there is a band on the surface that is completely free of intersections with other objects.



The highest setting of the slider is called Max, and it means Groboto will create quads that are as long as it is possible before running into the object end or an intersection with another object.

All these choices - mesh structure, variable density, long quads - could be also specified separately for each object. What you set in the Unified Mesh Output dialog box for all objects could be overridden from the Selected Object Properties tab (as above for density) for any selection of objects.



UniMesh Export Overrides: 52

☐ Dens 32

☐ Hi Cap 100

☐ Lo Cap 100

☐ Structure Relaxed

☐ Variable Density

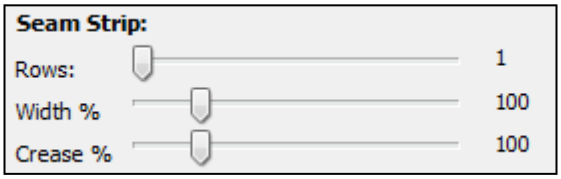
☐ Surface Row ☐

☐ Long Quads Off

For each property, if you want to override the global setting for the current selection, check the checkbox on the left side. For multiple object selections the checkbox might appear in undefined state - this means it's ON for some members of the current selection, OFF for others. Its state becomes defined once you click on it. Likewise, if the Structure popup is blank, it means currently the mesh structure choices are different for different objects in the multiple selection.

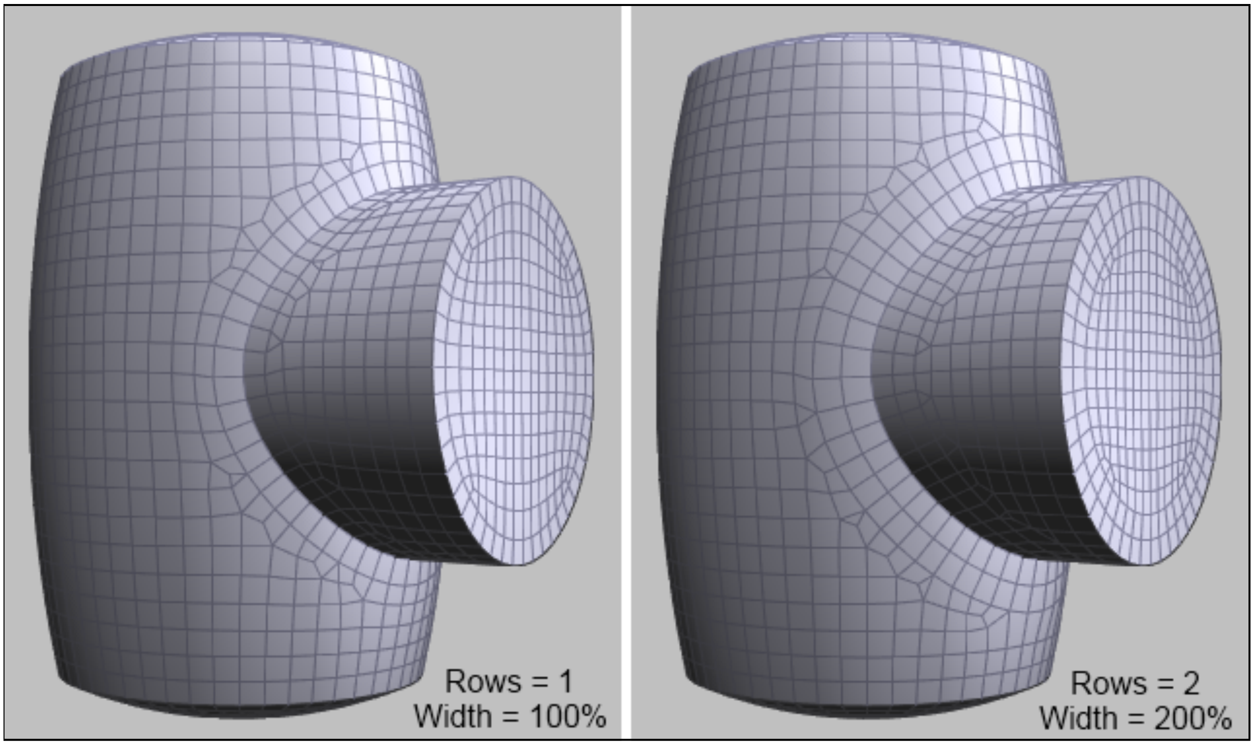
Seam Strips

The seam strips are very important in the Groboto unified mesh and they have their own set of special controls.

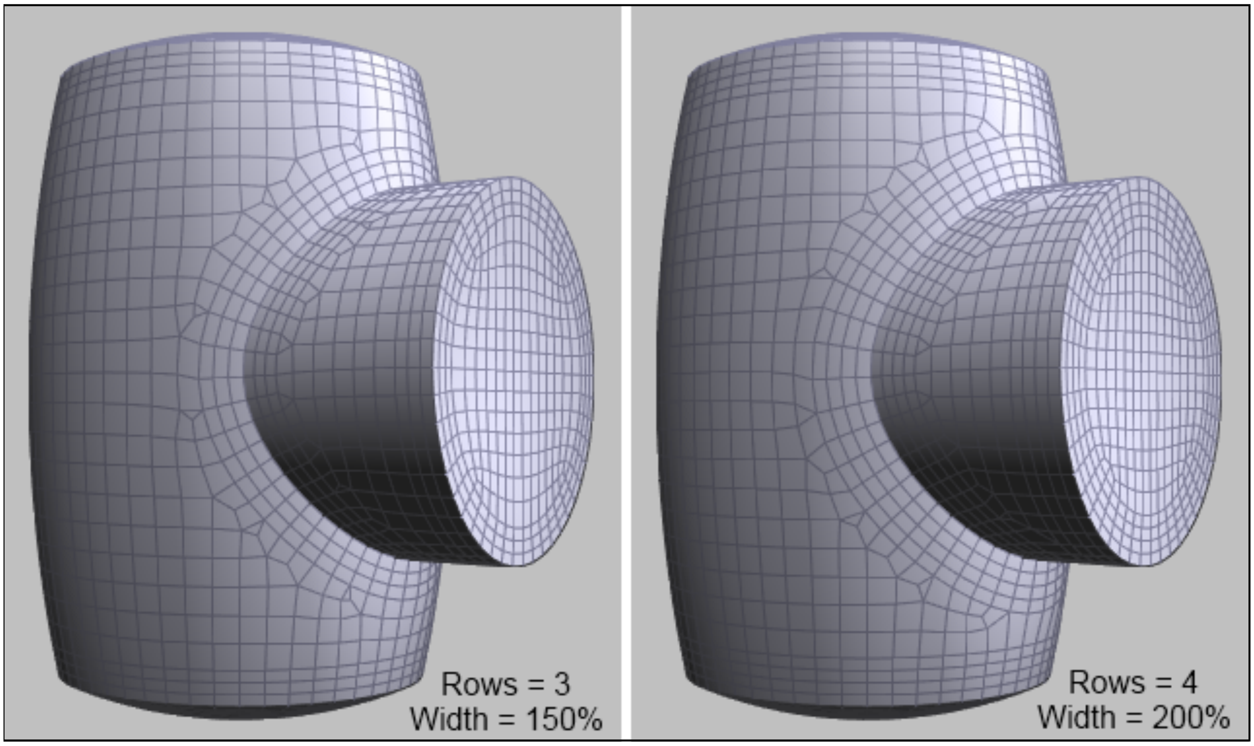

Seam Strip:

Rows:	1
Width %	100
Crease %	100

The strips consist of regular rows of quads along the seam intersection curve.



The number of rows can be set with the Rows slider (the number of rows is the same on both sides of the intersection curve). The width of the strip can also be set arbitrarily with the Width % slider. This width is expressed as a percentage of the quad length in the direction along the strip. For instance, if there is one row, setting Width to 100% produces square quads, width equal to length. If there are two rows, setting Width to 200% produces two rows of square quads. Any other Width setting creates non-square quads.

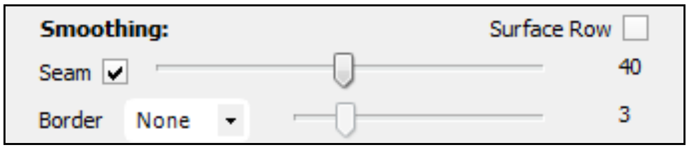


By default all rows have the same width. However, the width of the row closest to the seam can be
controlled separately, using the Crease% slider. When this percentage is less than 100, the row right
next to the seam is narrower than all the others. When it's more than 100, the row next to the seam
is wider than the other rows. Row width is redistributed without changing the overall strip width.

Seam Smoothing

All the mesh options we looked into so far produce a mesh describing exactly the same surface as we have in Groboto's non-mesh object-based environment. These surfaces have sharp seams where different objects meet each other (the surface is not smooth). Most of the time you'd be interested in creating smooth surfaces, where the sharp seams between primitive objects are rounded to a greater or lesser degree. The degree of rounding can be very different - from maintaining a pretty sharp edge to essentially blending the two objects into one, but some rounding is almost always desirable.

Groboto unified mesh, of course, is deliberately constructed so that sharp seams could be rounded easily and cleanly - by moving vertices in the seam strips (and nowhere else). This is done by checking the Seam checkbox in the Smoothing section of the Output dialog and setting its slider to some value in the middle.

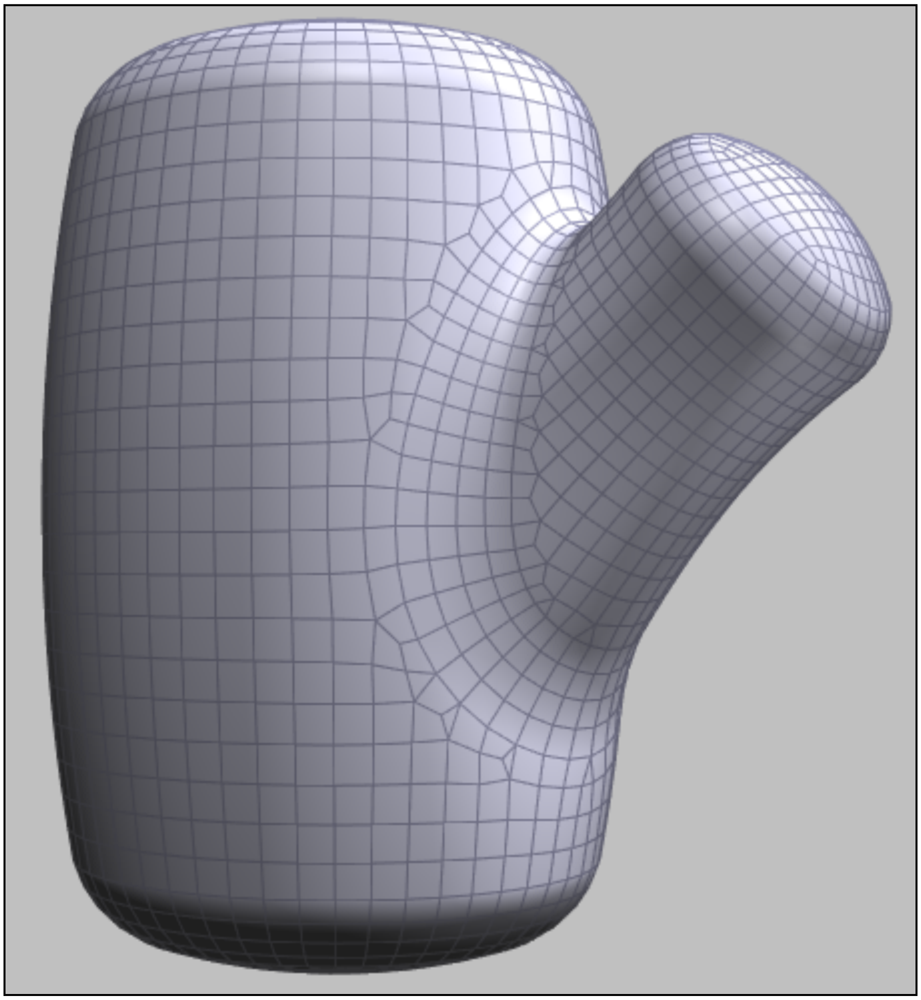


Smoothing: Surface Row ☐

Seam ☑ 40

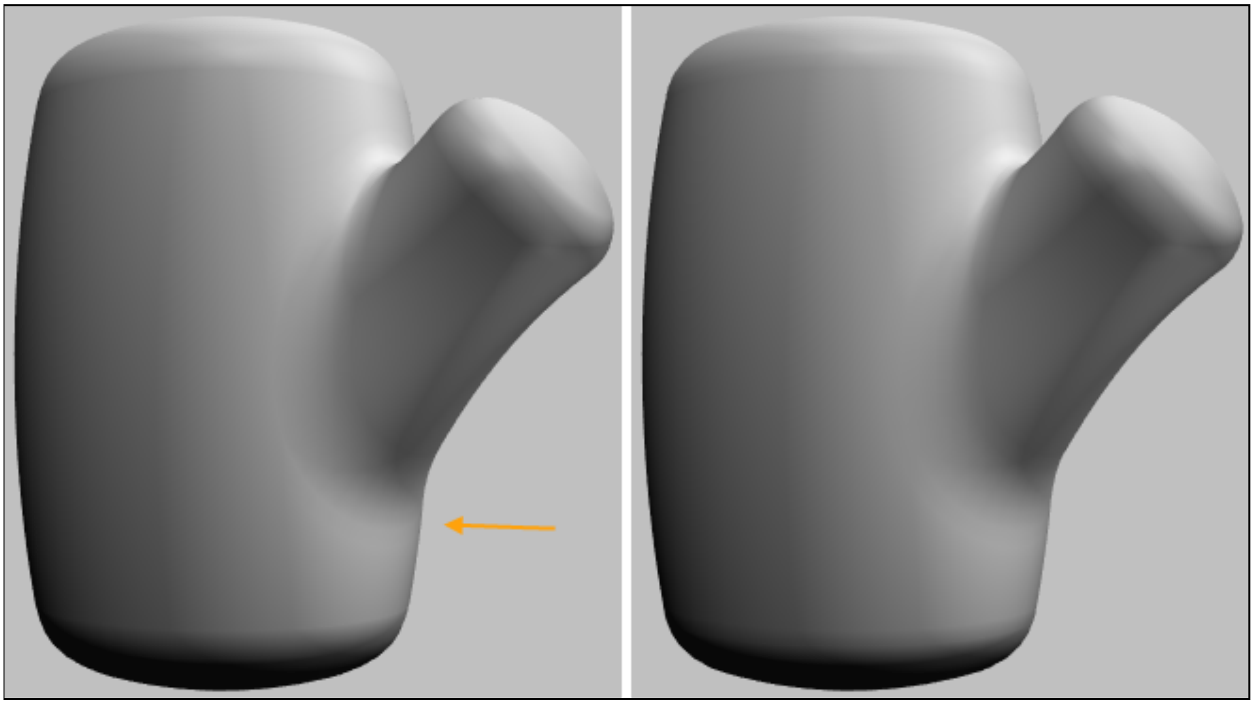
Border None 3

This keeps the outermost vertices of the strip (where it connects to the rest of the mesh) firmly in place, while the middle of the strip is allowed to relax and smooth out the sharp seam.



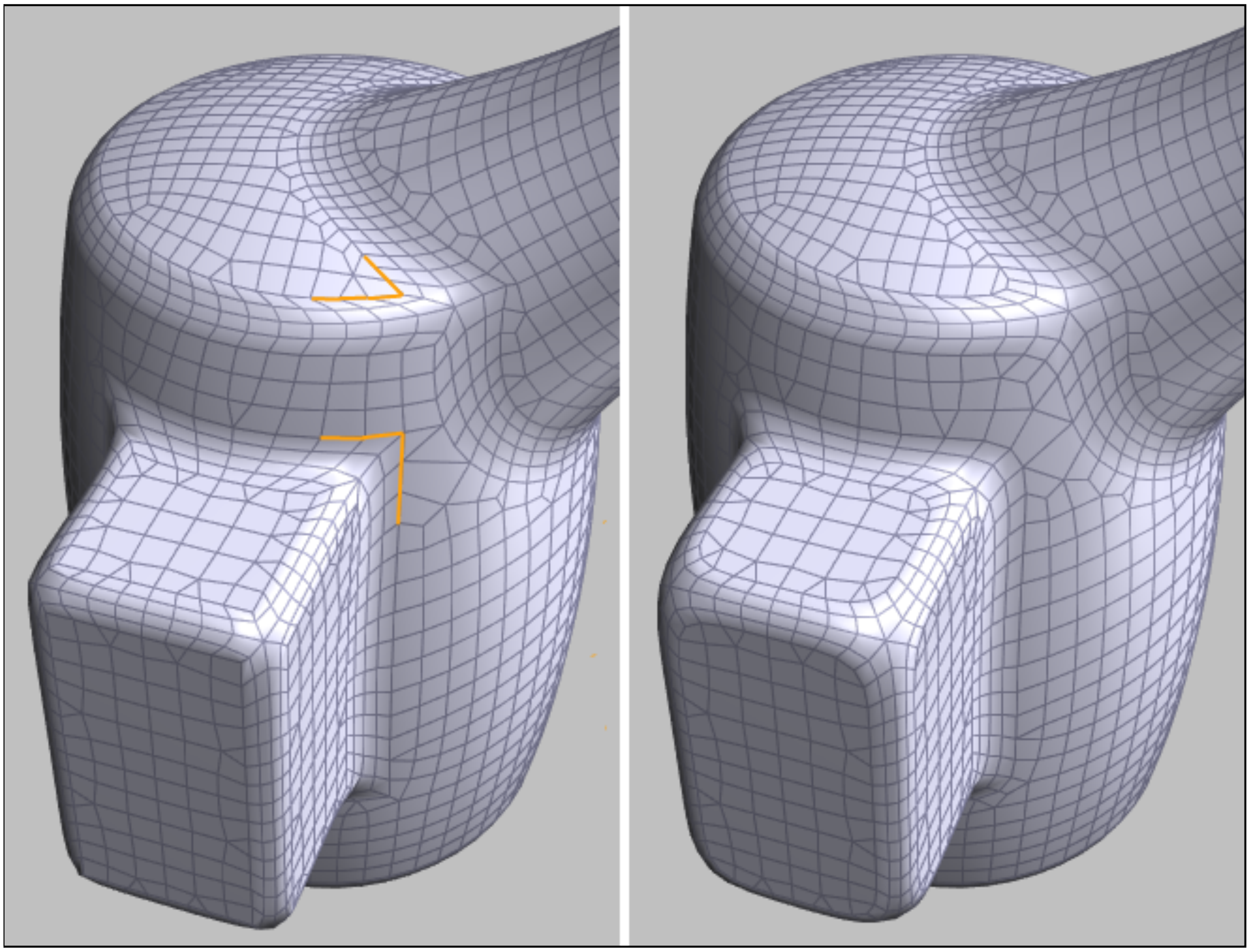
Rounding of the edge is always confined to the strip, thus the sharpness of the rounded edges is controlled mostly by the width of the strip (see above). For a given strip width you can still vary the rounding effect by moving the Seam slider. Smaller values produce only a slight smoothing effect - the seam becomes a little less sharp, while the rows in the strip farther away from the seam hardly move at all. Greater values spread the smoothing effect farther from the seam, producing more gradual rounding. At some point (depending on the number of rows) full rounding is achieved and moving the slider farther to the right has no effect.

Despite the fact that the seam strip blends smoothly into the rest of the surface, the boundary between them typically is quite visible. In the image below left, rendered in the smooth shaded mode, the orange arrow points to the edge of the strip. The human eye is very sensitive to shading patterns, and cannot be fooled by mere continuity of the shading value. If you are aiming for a more organic transition, a more subtle degree of continuity is needed. This can be achieved by checking the Surface Row checkbox (below right):



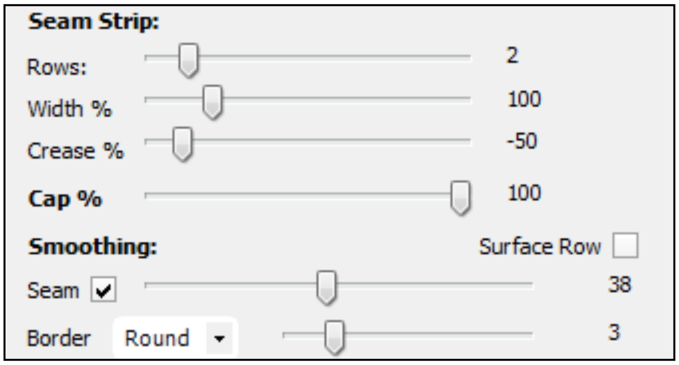
With this option the outermost row of the seam strip is not allowed to leave the original Groboto surface, but lingers near it, providing that smoother transition. Note that this effectively reduces the width of the strip by one row. Using this option also has a beneficial effect when the mesh is subdivided later in another 3D program, because it inserts one buffer row of quads between the area C where the mesh is somewhat disorderly and the inner part of the seam strip, where the surface could be very tightly curved. Without such a buffer you might see a string of gentle bumps or dents running along the seam in this area. The global setting for Surface Row can be overridden from the Selected Object Properties tab (as for density and mesh structure) for any specific object or a set of objects.

In most cases seam smoothing alone produces fairly awkward results at the corners, as seen here on the left:



This happens because seam smoothing allows the middle of the seam strip to deform, but keeps the boundary of the strip at a fixed position. This boundary is marked in orange in a couple of places (there are many other corners too). The strip itself wants to turn smoothly as it passes through the corner, but the sharp angular boundary does not let it. What's needed here is smoothing the boundary curve at the corners, before it is used as a fixed boundary for smoothing the strip. This is done by choosing Round in the Border drop-down and setting its slider, typically to some pretty small value (3 is used in the image above right). The boundary of the strip is rounded, allowing the deformed strip to flow very naturally through the corners.

When using the above described two options in the Border drop-down - None and Round - the already mentioned Crease% setting has a significant effect on the shape of the deformed seam strip. The examples above were produced using the default setting of the Crease% slider. To signify its new role when used with smoothing, as soon as you enable seam smoothing by checking Seam, the default Crease % value changes to zero. Zero means neutral, the plain vanilla rounding effect. When the Crease % slider is moved to the left, the values become negative:



Seam Strip:

Rows: 2

Width % 100

Crease % -50

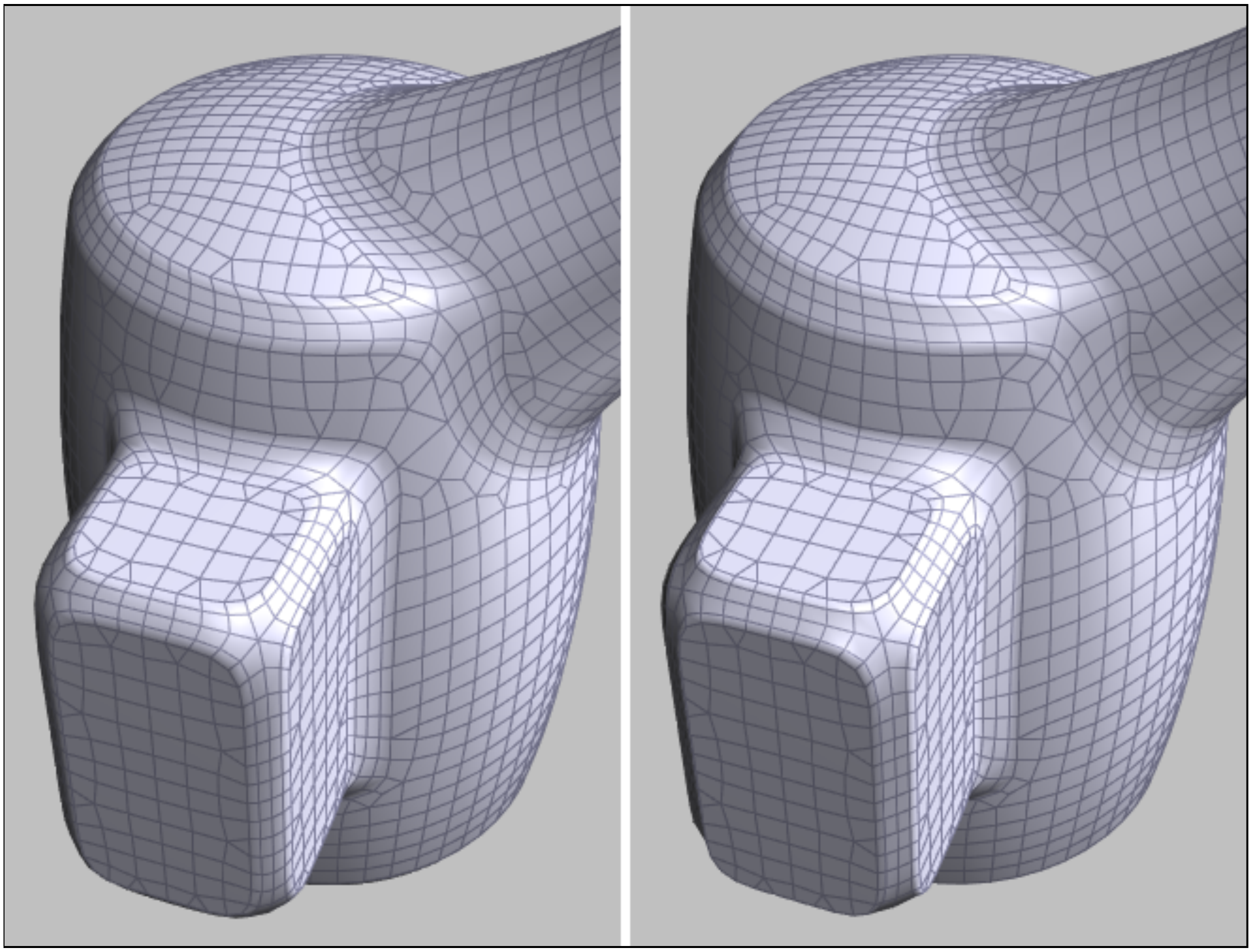
Cap % 100

Smoothing: Surface Row ☐

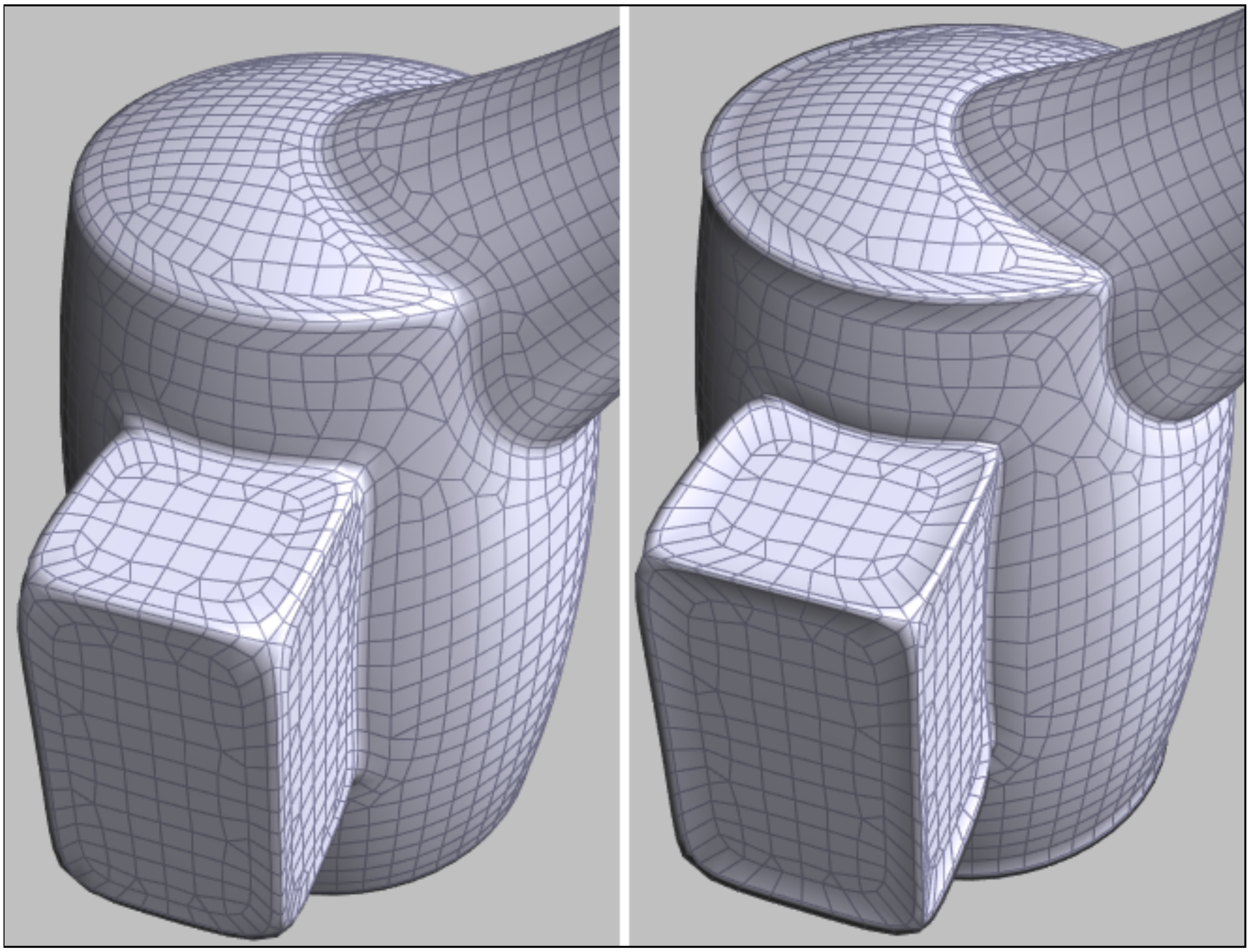
Seam ☑ 38

Border Round 3

and the rounded seam flattens to different degrees, becoming more like a bevel. Here we see the result with Crease % = -50 (left) and Crease % = -95 (right):



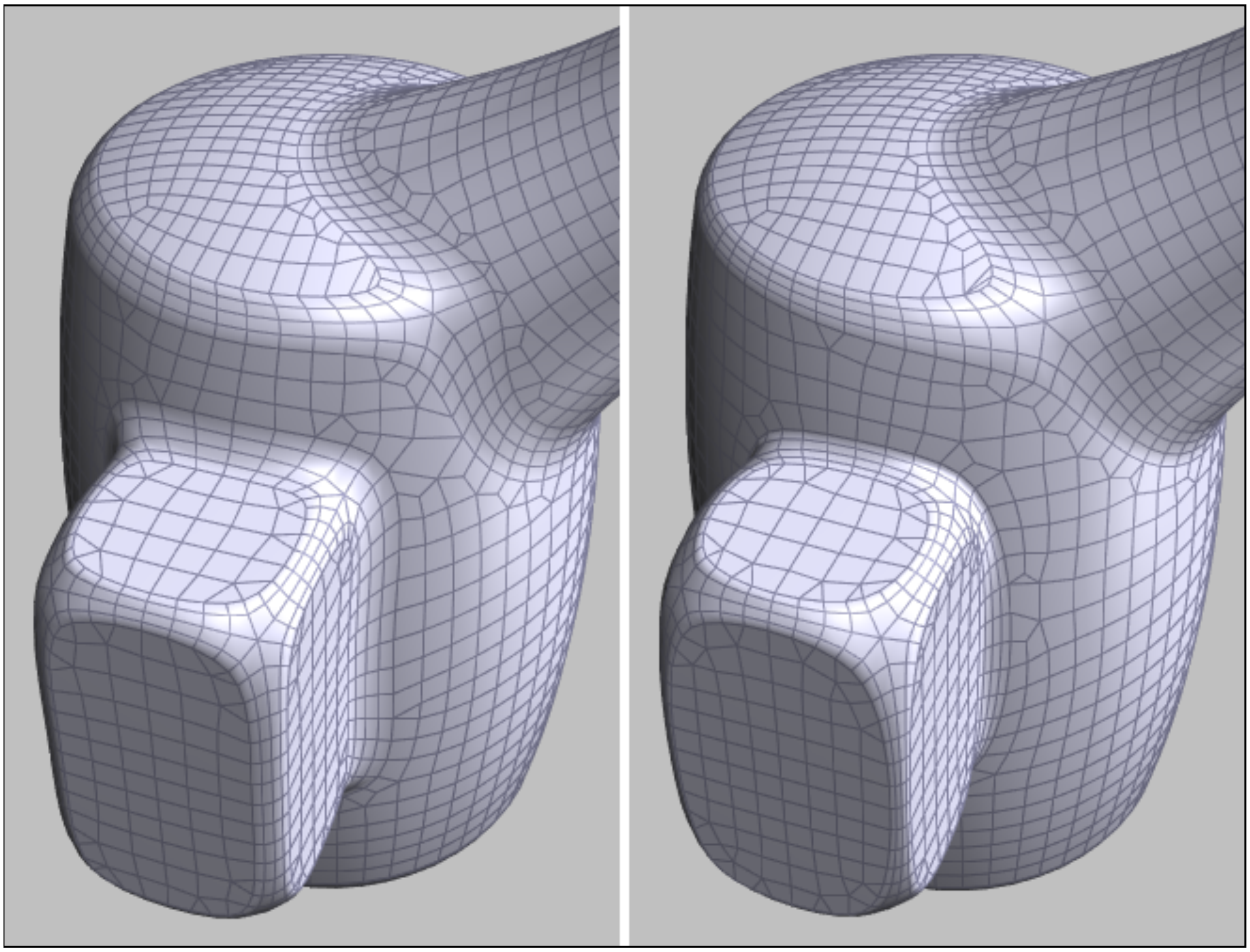
When Crease % slider is moved to the right the values become positive, and we have the opposite effect - the rows are squeezed towards the center of the strip, and for higher values they form a protruding ridge that rises even higher than the original sharp edge. Below are the results at Crease % = 150 (left) and Crease % = 300 (right):



An alternative approach to achieve natural seam smoothing near the corners is to use the Drift option in the Border drop-down. Rather than smoothing the strip boundary as a separate procedure before smoothing the seam strip, this goes ahead with strip smoothing right away, but without holding the strip boundary to its original position. The strip boundary freely slides along the surfaces on both sides of the seam (but it cannot leave the surfaces) and the boundary itself is rounded at the corners as the result of general strip relaxation.

With this method Crease % and Border sliders have no effect. The Seam slider (the degree of relaxation) has very significant, and open-ended, effect. It was mentioned above that with other methods (None and Round), as you move the Seam slider to the right, eventually full relaxation is achieved and nothing changes any more. With Drift as you move the Seam slider to the right, the strip vertices drift more and more from their original positions, with no limit.

For smaller Seam slider settings the results of this single-stage Drift relaxation could be very similar to those achieved by the above method of first rounding the boundary, then relaxing the strip to conform to the boundary. With higher settings the results differ significantly (in this example Seam = 40 on the left, Seam = 65 on the right):

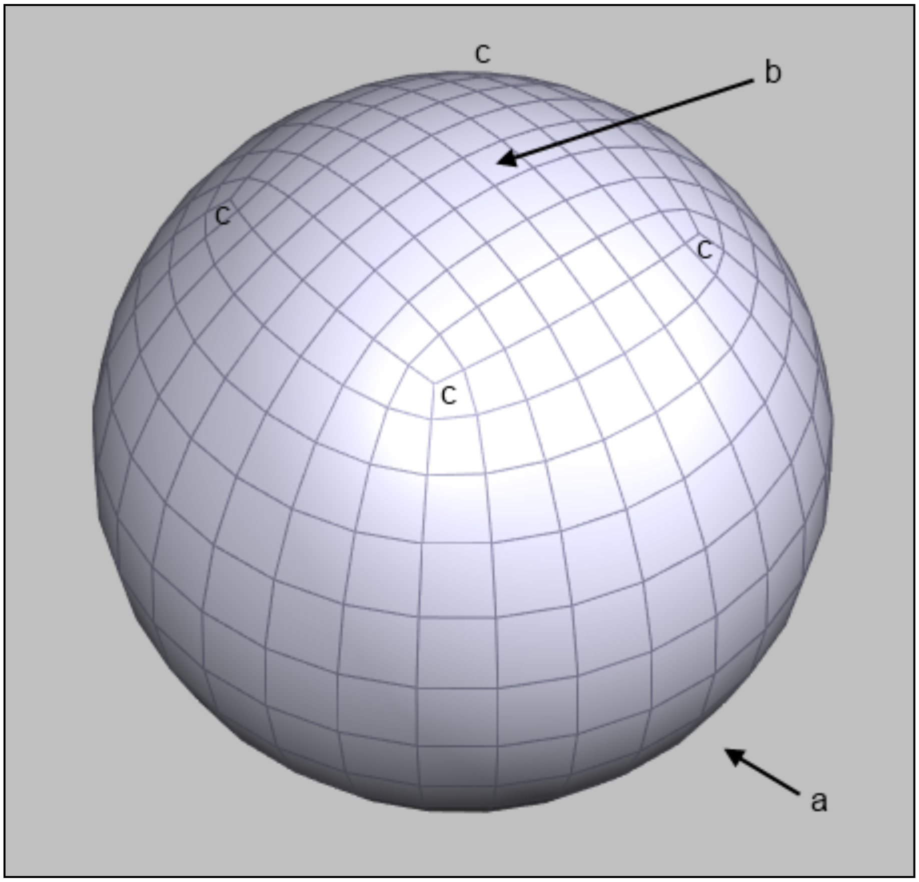


The main difference is that here nothing holds the strip boundary to its original shape - for instance, the top face of the attached box essentially became round. Typically with this method the width of the relaxed strips varies widely.

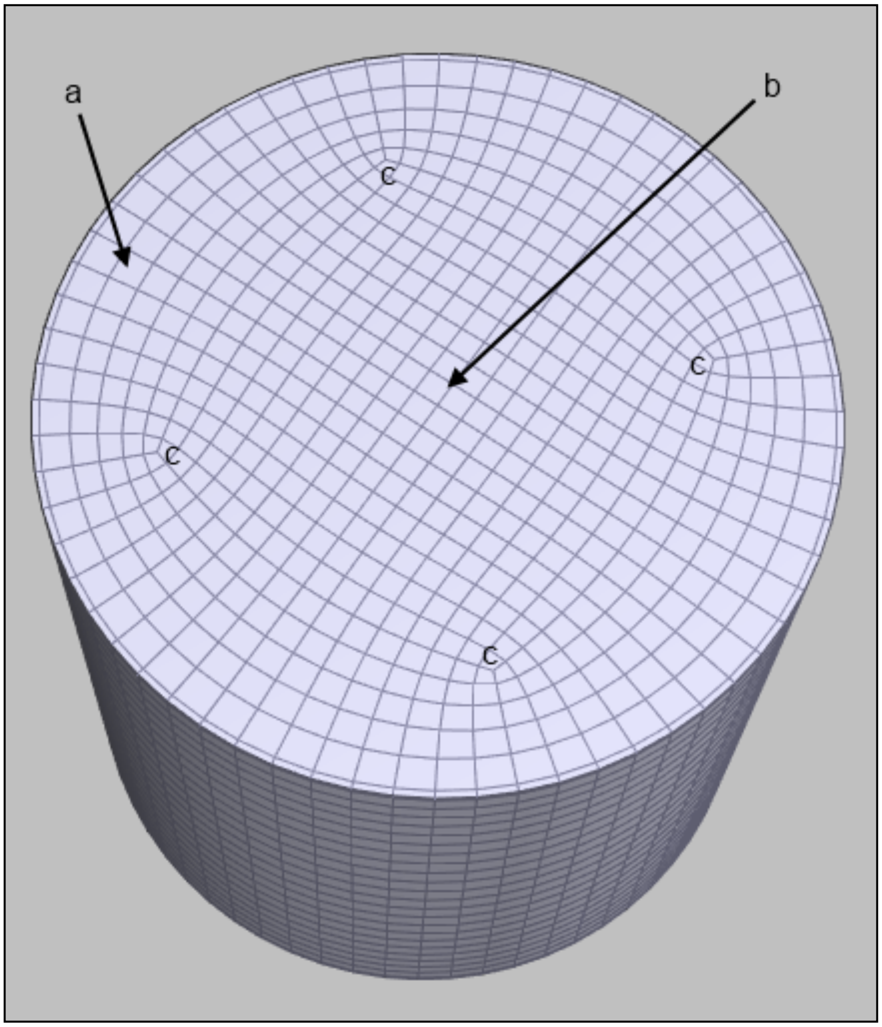
Finally, choosing Both in the Border drop-down combines both methods - the strip boundary drifts freely, but from a starting position defined by preliminary border smoothing, controlled by the Border slider.

Polar and cylindrical mesh

The native mesh on many Groboto primitives consists of several distinct parts. Sphere is a good example:

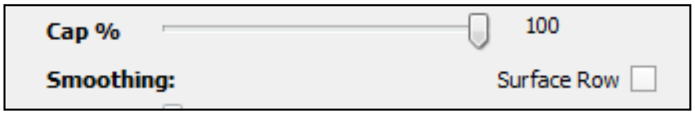


Most of the mesh on a sphere has cylindrical structure (a) – organized like mesh on a sidewall of a cylinder, with rows along the circumference (equator) and columns along the axis. We could have continued this type of mesh organization all the way to the poles, but this would produce narrower and narrower quads as we get closer to the poles. Unlike with most other meshing solutions, in Groboto any piece of native mesh has to be ready for possible interaction with seam strips, if our object happens to intersect in space with other objects. It is for this reason that we try to keep native mesh quads in Groboto as square as possible – nearly square quads behave much better when native mesh has to be sewn to seam strips. Because of this, cylindrical mesh structure is always kept some distance away from the poles; near both the poles mesh switches to square polar organization (b).



The same happens on the caps of cylinders and other similar objects. Towards the rim the mesh on a cylinder cap is cylindrical in structure (circular rows and radial columns), but closer to the center it switches to square polar organization (b). On a sphere or cylinder cap the boundary between the cylindrical mesh and the polar mesh is indicated by the four special vertices (c) that have only three edges connected to them, rather than the usual four.

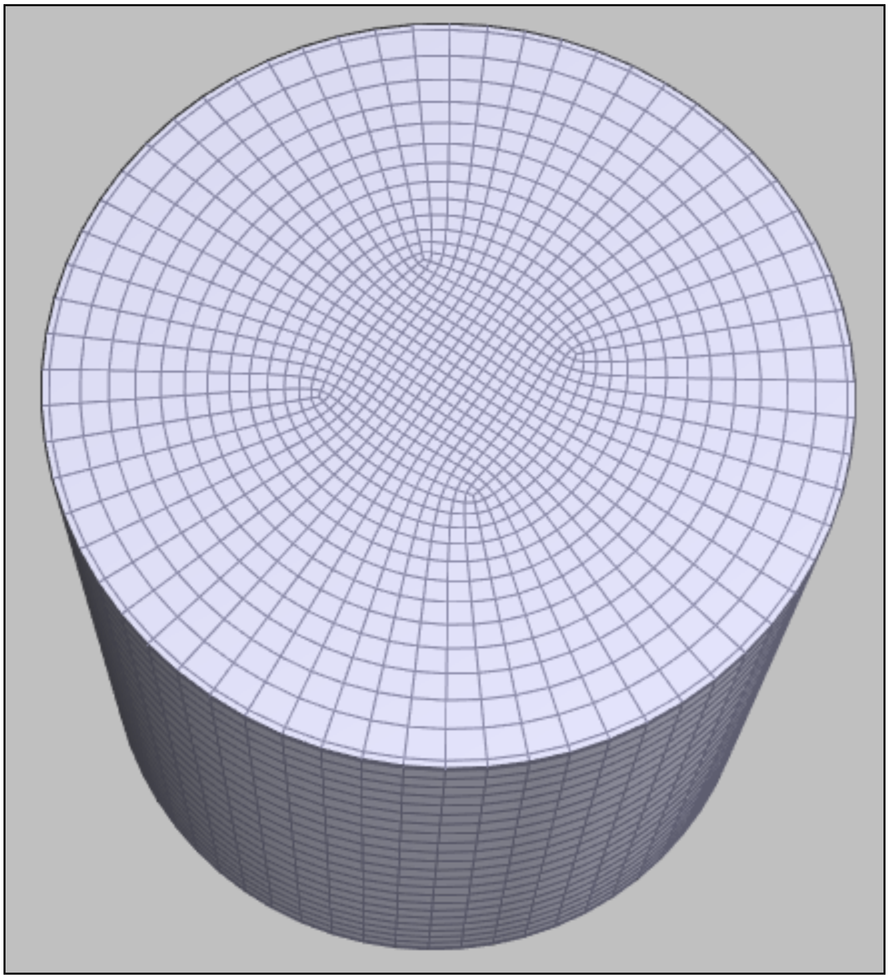
Objects of these types always have cylindrical and polar mesh in Groboto, but you can shift the boundary between the two types of mesh using the Cap Percentage slider:



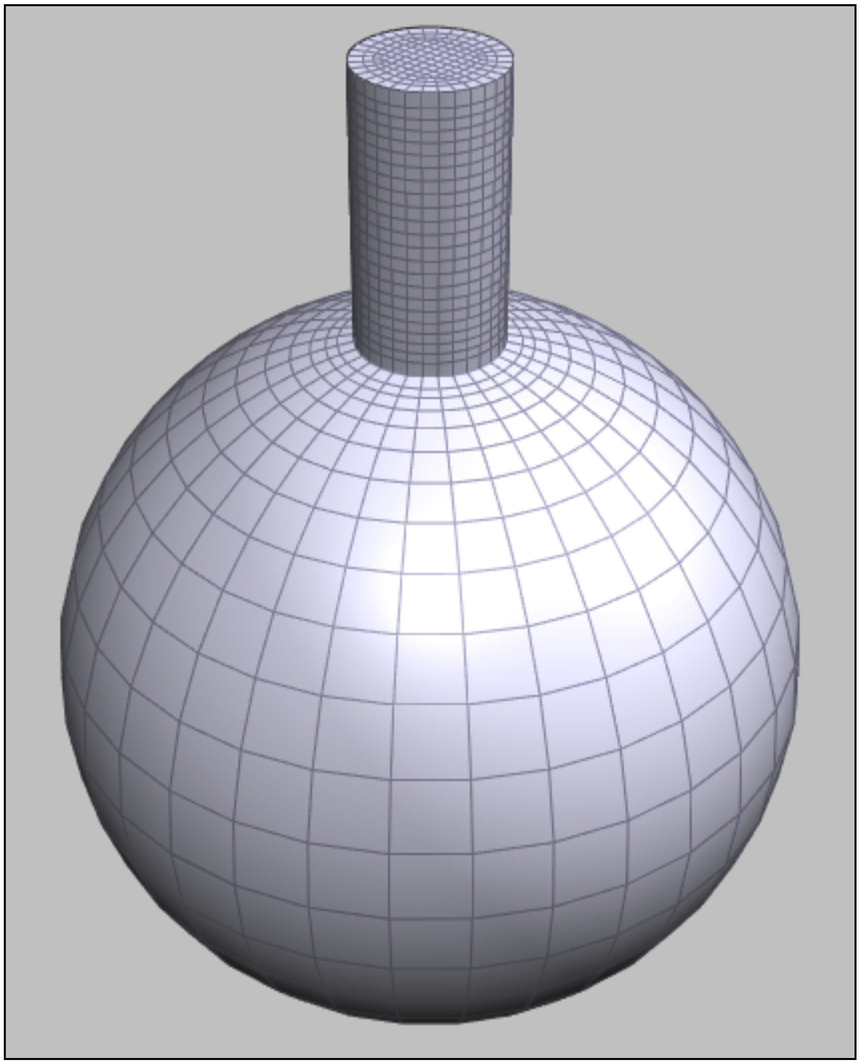
Cap % 100

Smoothing: Surface Row ☐

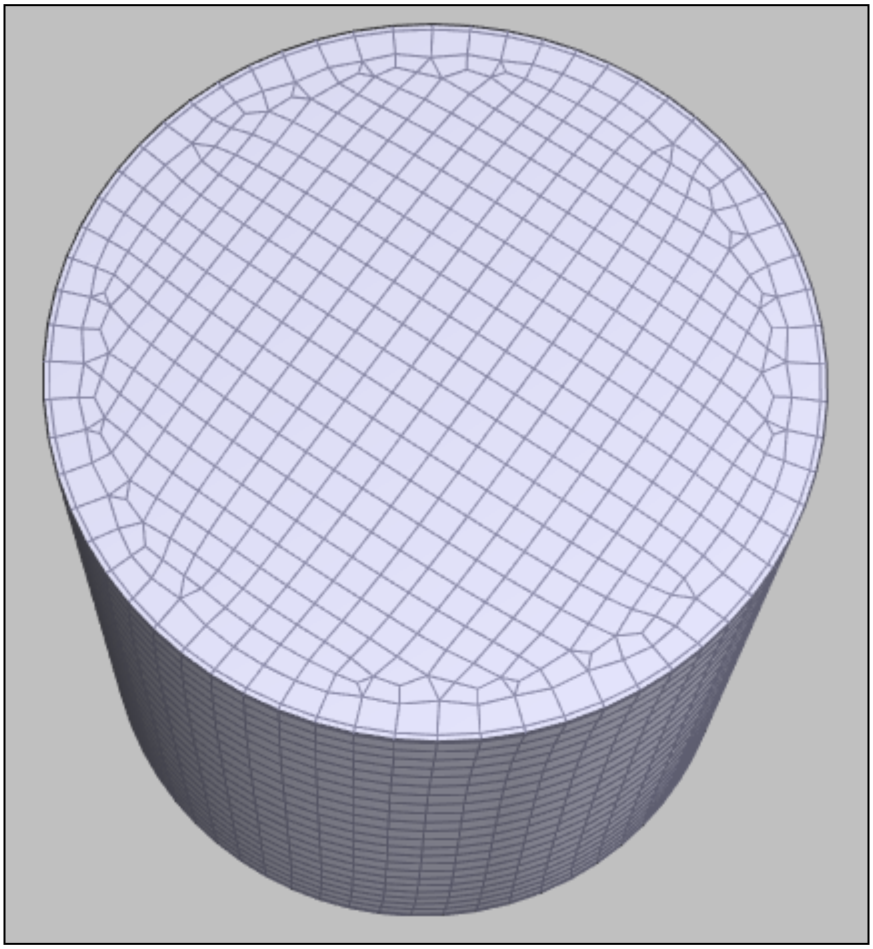
The default is 100% - the largest polar mesh area that you can have. Reducing this value shrinks the polar mesh area and expands the cylindrical mesh area (Cap % = 40 in the image below):



On an isolated sphere or cylinder there is no advantage to shrinking the polar mesh (the total number of quads only increases). However, if some other object is attached to the polar area of a sphere or the middle of the cylinder cap, you can shrink the polar mesh so much that only cylindrical mesh remains on the used part of the surface, which creates a much cleaner mesh.



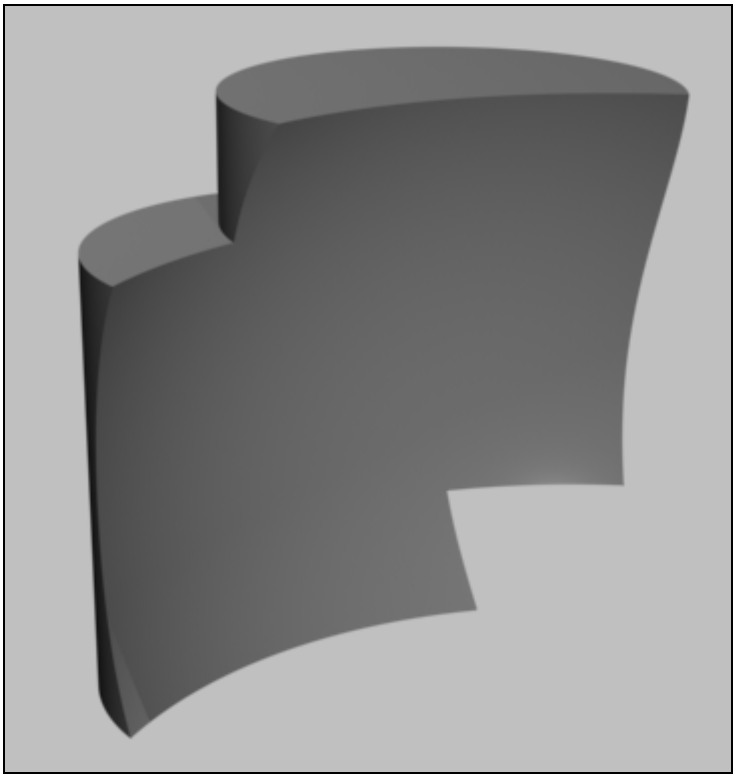
On cylinders a special setting Cap% = 1 generates a simple uniform square mesh pattern on the cap, like the one used on square sides of a cube. Usually a pattern like this is not desirable on the cap of a cylinder because of its irregular connection to the circular edge of the cap, but in some cases it is advantageous.



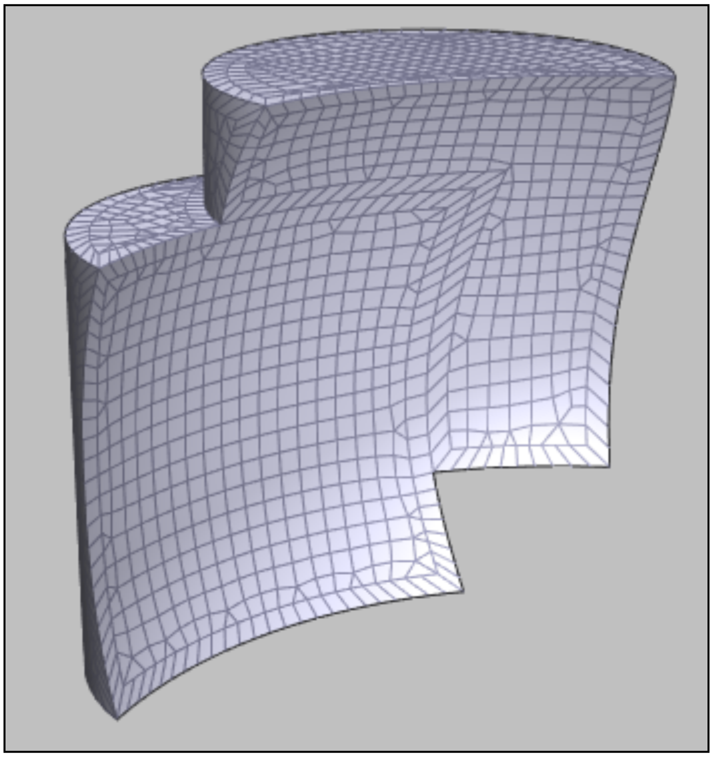
Just like with mesh density, setting Cap Percentage in the Unified Mesh Output dialog box affects all the objects in the scene, but you can override this setting for individual objects from the Selected Object Properties tab of the Main Groboto panel. With one or more objects selected, check the Hi Cap or Lo Cap checkbox (cylinders, cones and sliced spheres have two caps and the Cap Percentage can be set independently for each cap) and set the slider as desired. For spheres, ellipsoids and hyperRods only the Hi Cap setting is active, affecting both caps - because for these objects both caps are on the same surface.

Shared Surfaces and Optional Seams

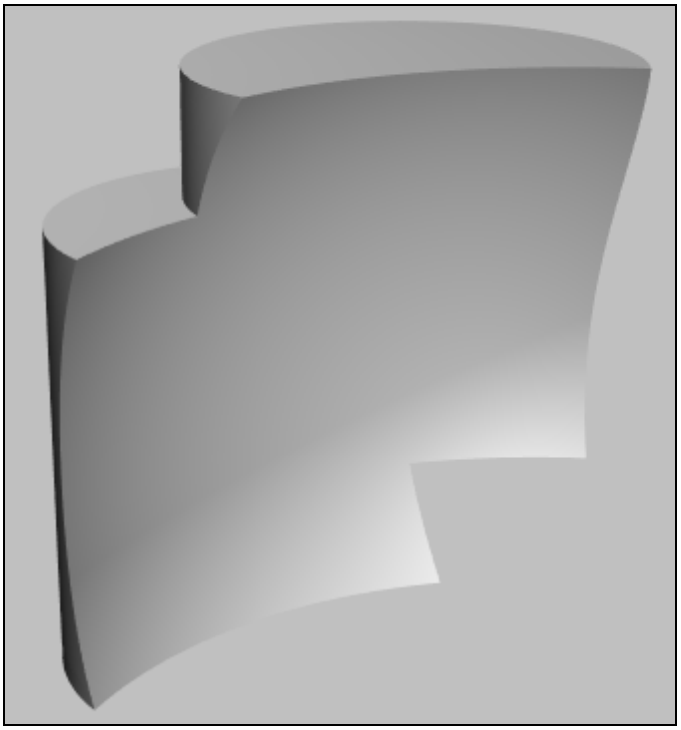
Some of the seams that appear in Groboto unified mesh are not really seams between different surfaces, but different instances of the same surface. Most often this happens when there is a boolean cluster with two or more primary objects, trimmed by a common trim object. In the example below we have two overlapping primary cylinders, trimmed by a subtractive sphere.



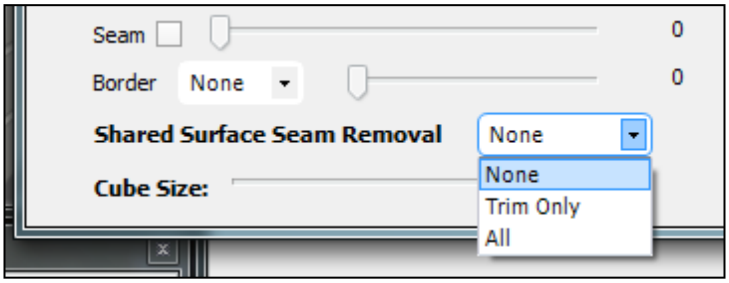
By default, the unified mesh of this cluster looks like this:



There is a seam where one of the cylinders ends and the other one begins, although on both sides of the seam it is the same surface (the sphere). Although the presence of the seam is quite obvious in the wireframe view of the mesh, if we render the mesh in smooth shaded mode, we won't see it - the shading is the same on both sides.



Nevertheless, whether or not the seam is present on a combined surface like this could be quite important if you plan to modify the mesh after exporting it from Groboto. You can choose not to have the seam, by choosing Trim Only in the Shared Surface Seam Removal pop-up:



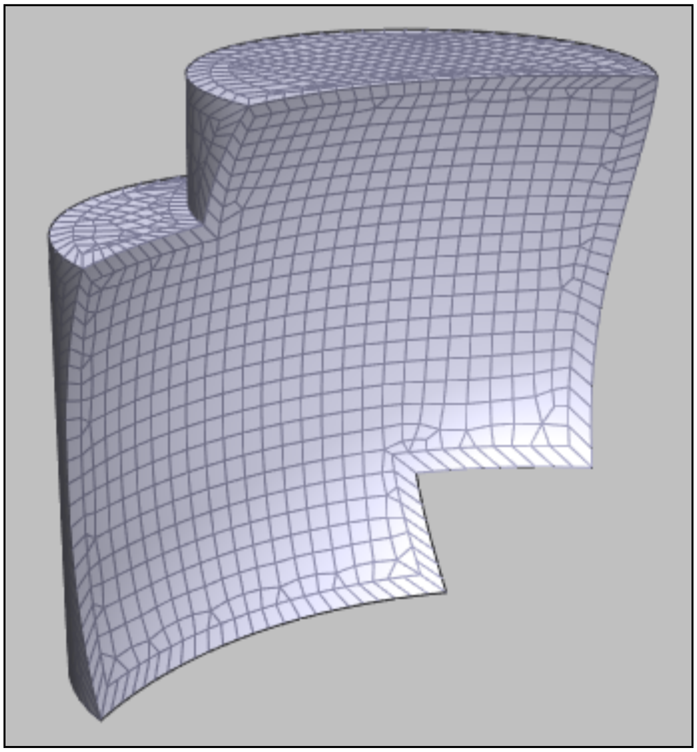
Seam ☐ 0

Border None 0

Shared Surface Seam Removal None

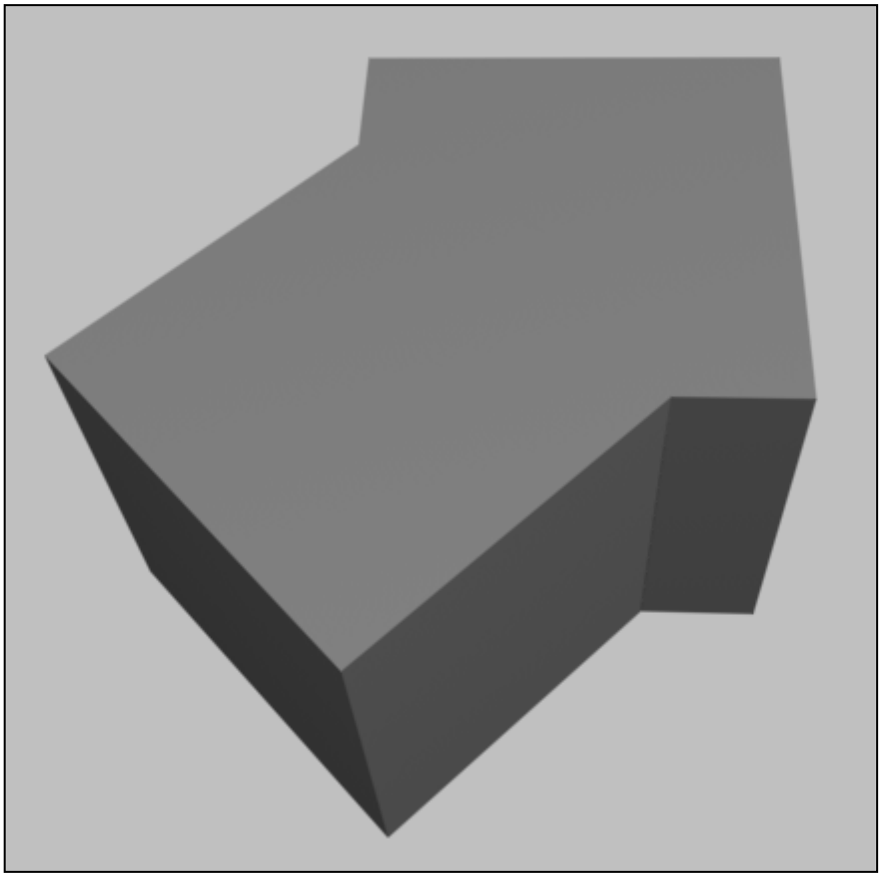
Cube Size:

The result is a single uninterrupted native mesh on the entire surface:

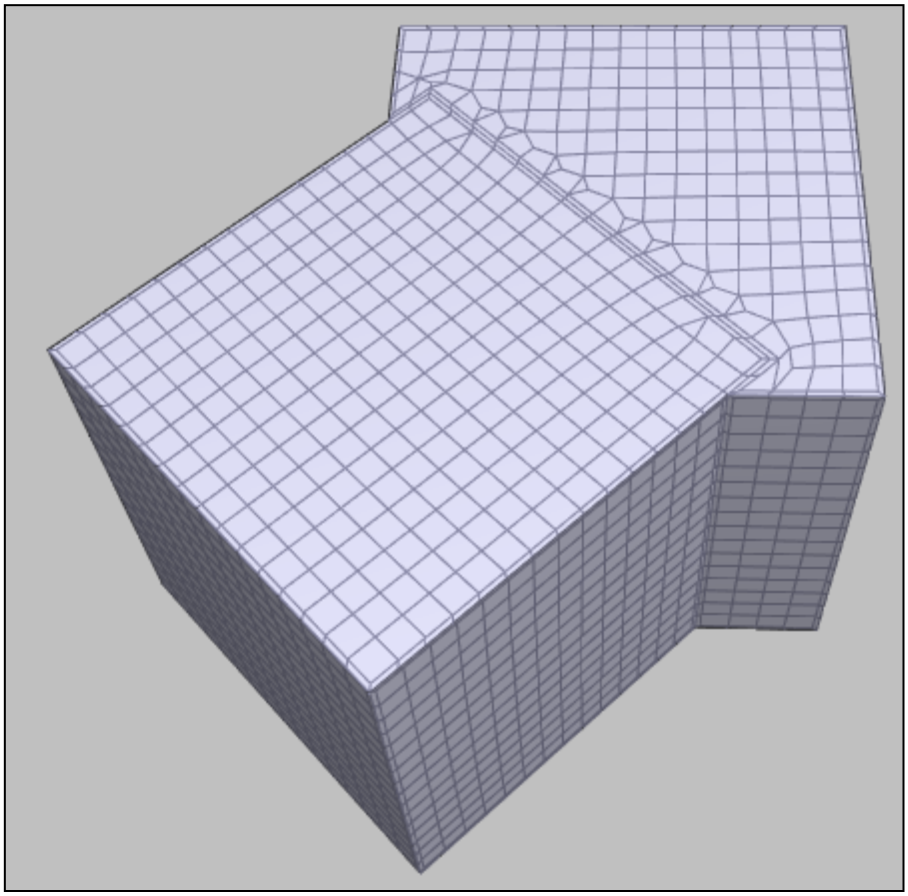


This flavor of seam removal (Trim Only) is always safe, because the native mesh here is already defined the same way (density, direction) on both sides of the seam, since it comes from the same trim object.

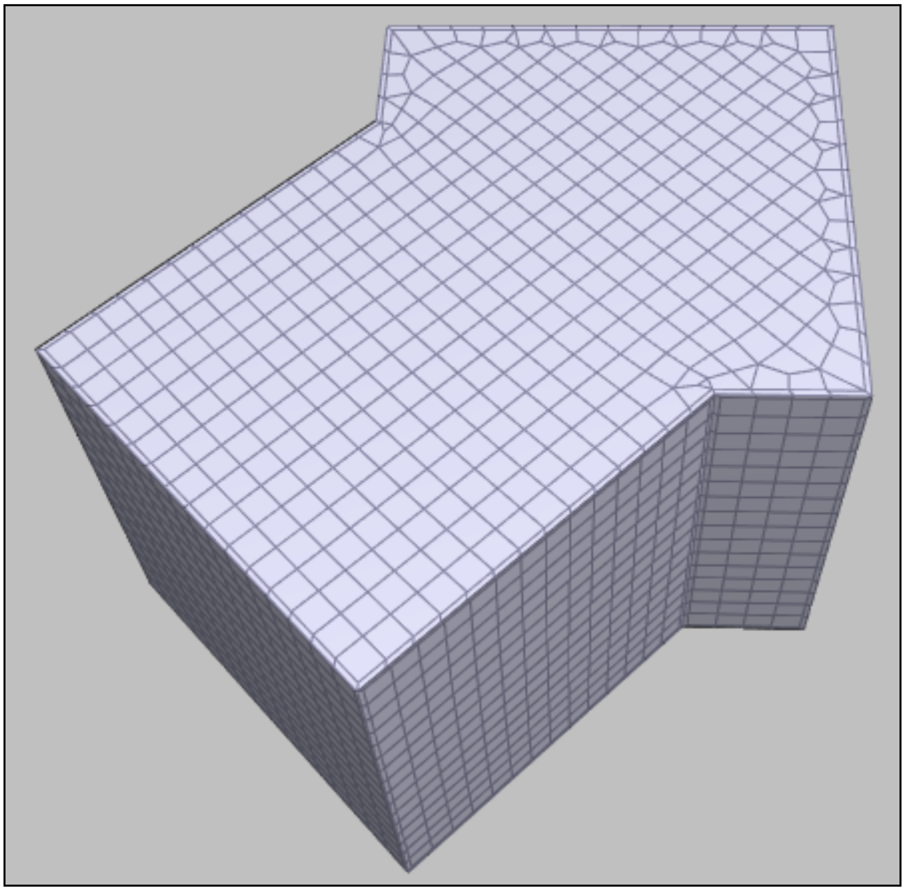
There is another situation in Groboto where we encounter a seam between identical surfaces, but in this case the mesh on these identical surfaces can be quite different. If we take a box, duplicate it, then move the duplicate along the top side, the top sides of the two boxes remain coincident. We can also rotate one of the boxes, just to make things more interesting:



By default (when None or Trim Only is chosen in the Shared Surface Seam Removal pop-up) Groboto generates the following mesh:



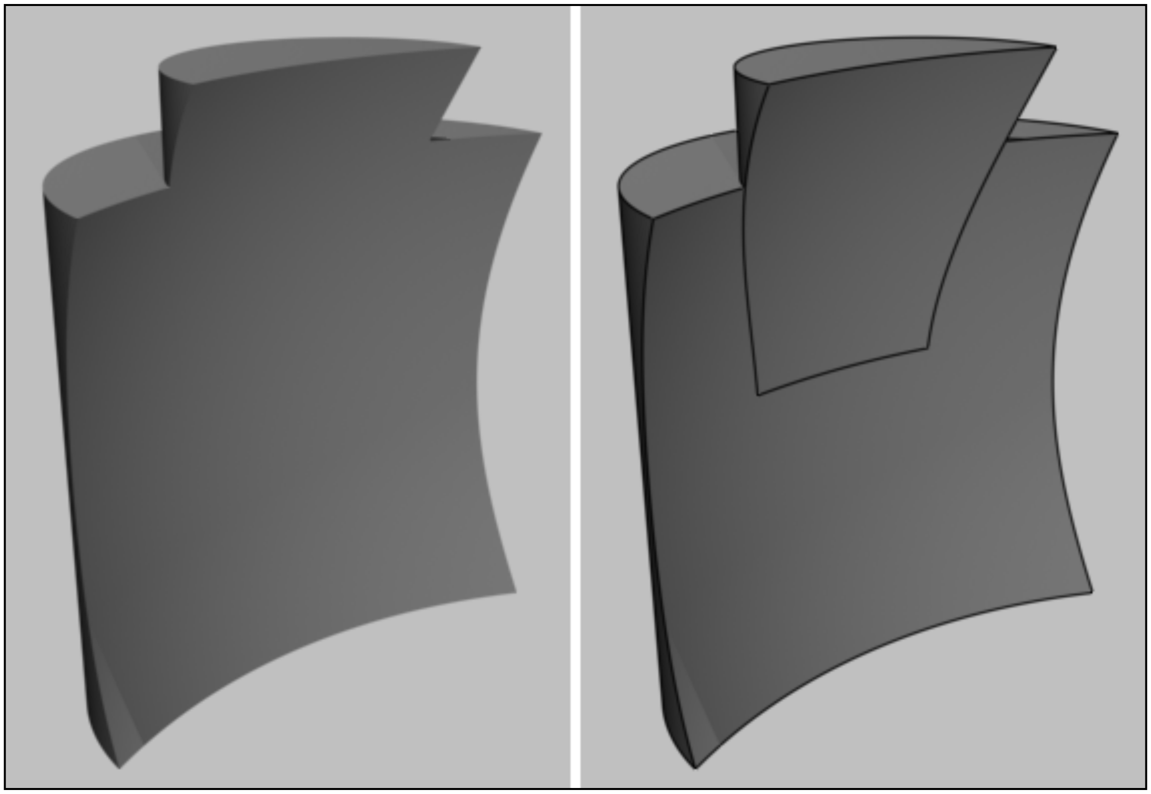
Here we might also want to get rid of the seam on the top side and generate one continuous clean mesh on the entire flat surface. However, unlike the case with the boolean trim, here the meshes are different on different parts of this surface (because they come from different objects) and they do not flow seamlessly into each other. We might even say that in this case the seams, disruptive as they are to the mesh, provide a useful service - by at least providing some method for stitching together of unrelated meshes. Still, we can make Groboto create a single continuous mesh on the entire surface by choosing All in the Shared Surface Seam Removal pop-up:



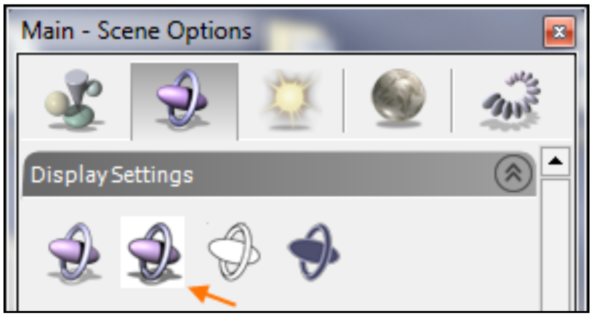
Groboto simply extends the mesh structure from one of the boxes onto the entire combined surface.

Optional Seams as Modeling Method

Just from the perspective of creating a simpler, cleaner mesh removal of optional seams is almost always desirable. But there are other perspectives. Having a gracefully curved seam running in the middle of some surface and dividing it into two or more patches provides a basis for a very powerful and elegant method of reshaping models once they are converted to mesh and exported from Groboto. Let’s use a somewhat modified - more elegant and symmetric - version of our first example, a Boolean cluster with two primary cylinders trimmed by one sphere (left):

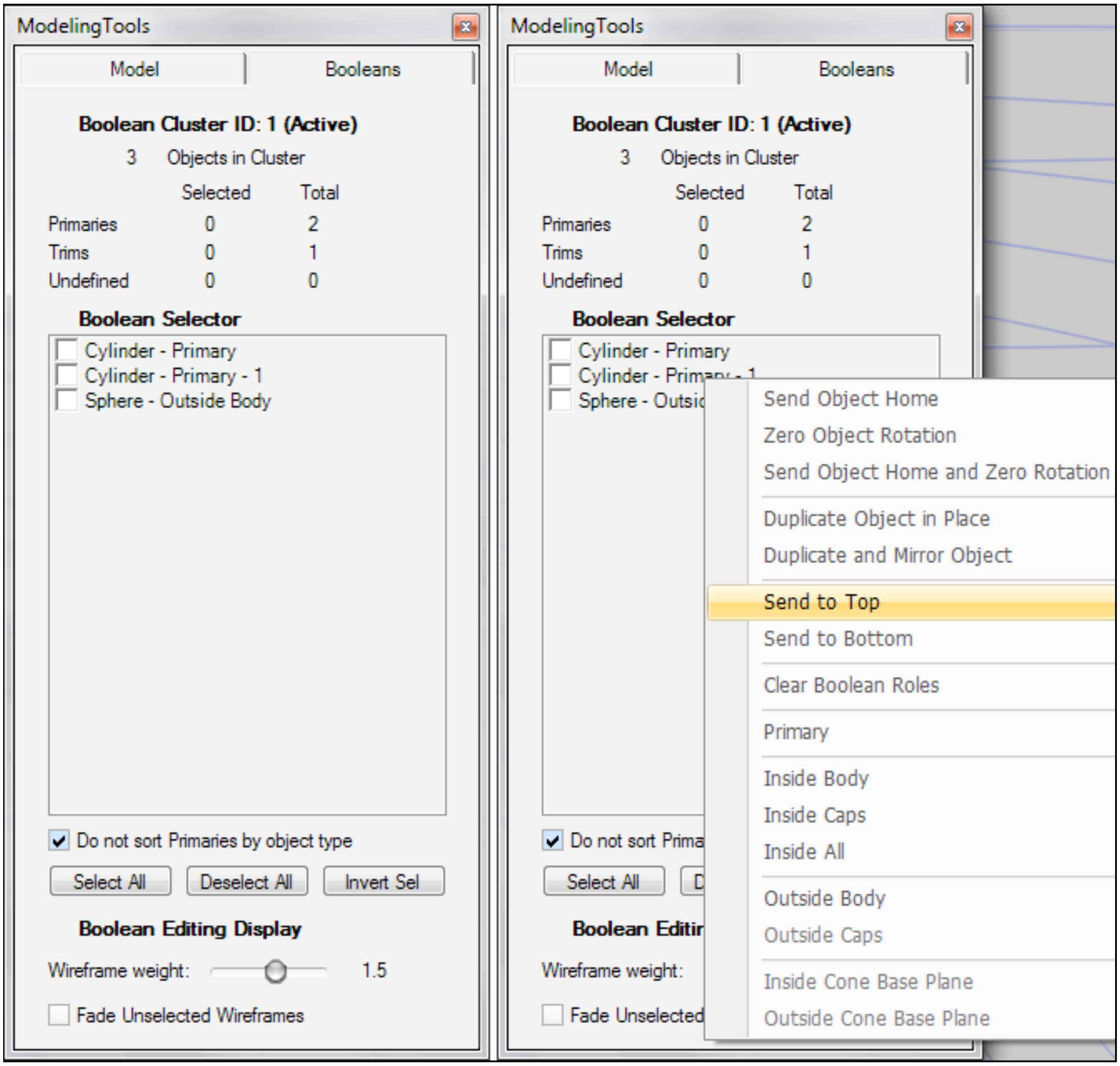


Here the top cylinder is smaller than the bottom cylinder. Groboto has a display mode that shows optional seams even before the model is converted to mesh (above right). It's called Shaded with Outline:

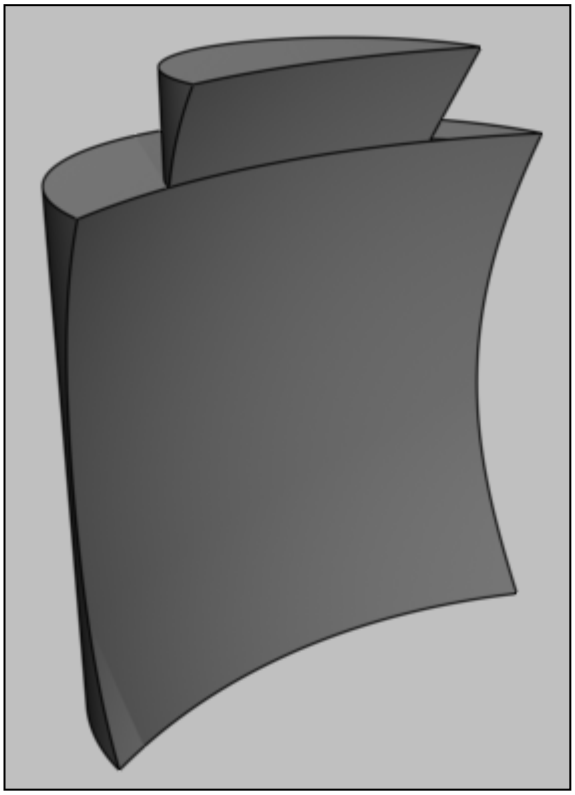


The Quick Line display mode (rightmost) also makes optional seams visible. In either of these two modes you can see the seams change in real time as you move and reshape the primitive objects. With the seam visible we clearly see the structure of this cluster - a smaller cylinder sunk into the larger one, and then the whole thing sliced with a sphere.

Note that the two cylinders are not treated equally: the lower part of the top cylinder, even though it's inside both cylinders, visually appears as belonging to the top cylinder. It's as if the part of the front surface that is inside the top cylinder is slightly raised above the rest of the front surface. We can think of multiple primary objects trimmed by a common trim as a stack of 2D layers composited into a single image: the top layer obscures all the other layers where it's present; outside of it we see the second layer, which obscures all the layers under it; and so on. The order of these 'layers' can be set arbitrarily, in the Boolean tab of the Modeling Tools panel (Window->Modeling Tools):

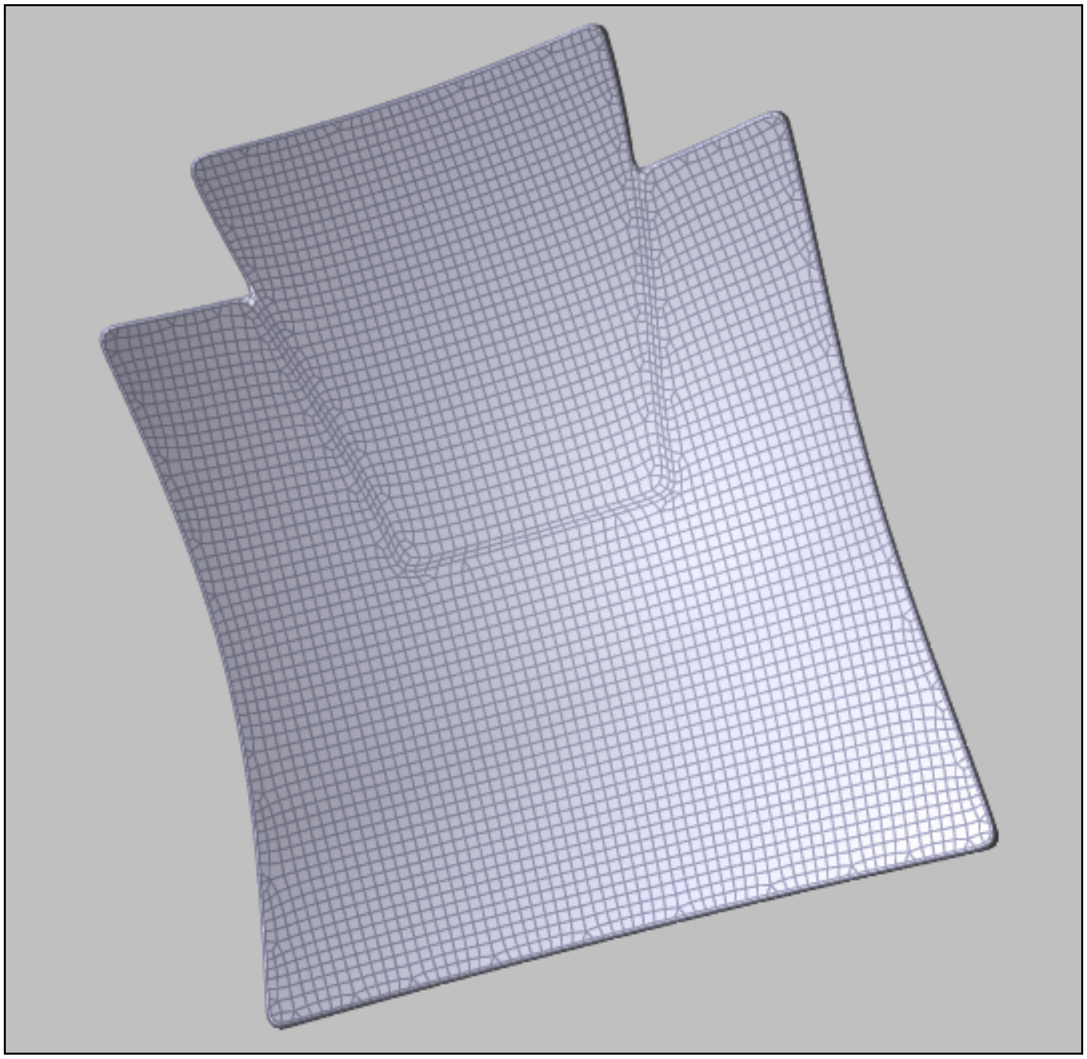


On the left we see the list of objects in our boolean cluster: the small cylinder at the top (Cylinder - Primary), then the larger cylinder, and finally the trim sphere. The top-to-bottom order in the list corresponds to the top-to-bottom order of the 'layers'. The small cylinder at the top appears 'raised' over the larger cylinder. If we right-click on the second (larger) cylinder and choose Send to Top in the menu, we'll get the following result:

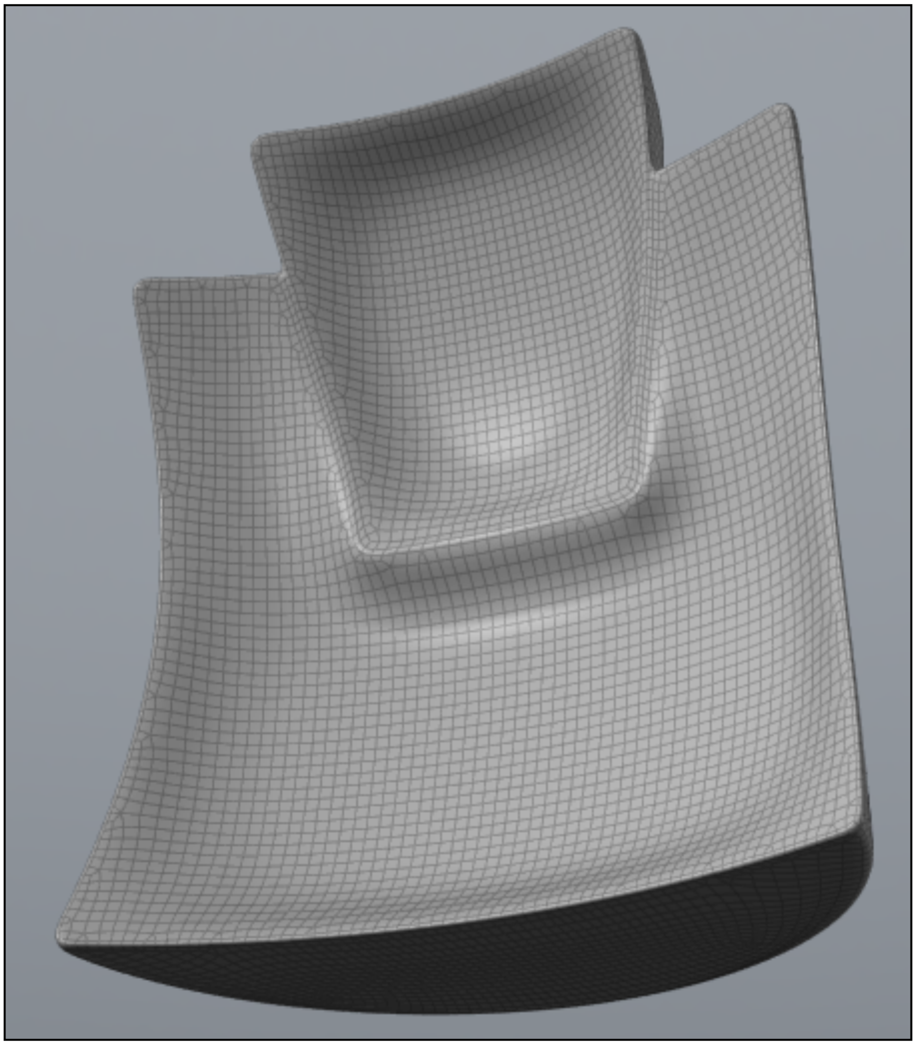


Now the larger cylinder is 'raised' above the smaller one, which creates a very different optional seam. Using Send to Top and Send to Bottom on various primary objects (or selection sets) any arbitrary order can be arranged. Note that when primary objects of different types are used (cylinders, spheres, etc.) it is necessary to check the 'Do not sort Primaries by object type' checkbox under the list - otherwise sorting by type takes precedence and you won't see the correct top-to-bottom order. It does not matter in our example here because both primaries are the same type.

Going back to the smaller cylinder on top configuration, if we convert this model to mesh (with Shared Surface Seam Removal set to None), we get this mesh:

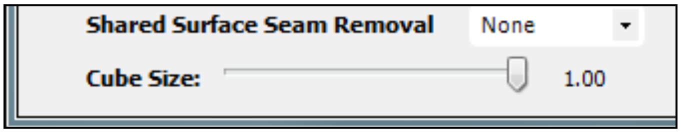


As unmodified Groboto model, it does not benefit from the presence of this seam; as we saw above, the seam is completely invisible when the model is rendered in smooth-shaded mode. However, as we are going to discuss below (page xxx), Groboto creates not only the mesh itself, but also the accompanying data (in the form of vertex weights or bitmap masks) that allows you to do mesh editing in some other 3D application, while still having access to the Groboto object level data such as patches and seams. Any deformation of the model could be made ‘seam-aware’, for instance we can make some patches shrink inward while keeping the seams separating them stationary. Here’s how it looks when we export the above example with a vertex weight map into Modo and apply a simple Move deformation to the mesh:



Only the surface away from the seams is pushed inward, the seams do not move. Here the seam that was left in the middle of the trim surface matters very much, we would have gotten an entirely different result if we removed it. This is one of the distinct Groboto-based mesh modeling techniques, where object manipulation in Groboto is used not so much to define the shape, as to create an elegant network of curves in space which guide subsequent mesh deformation with other tools.

Cube Size


Shared Surface Seam Removal None

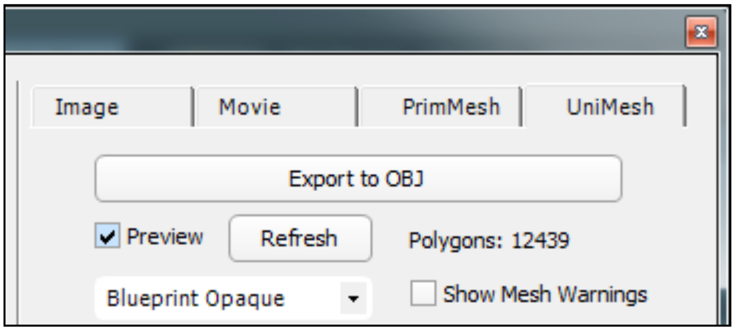
Cube Size: 1.00

This slider is used very rarely, to address a rarely encountered problem when the unified mesh produced from a model has some small parts of the model missing (it can also show up as a message "Cannot generate mesh", creating no mesh at all). To produce unified mesh Groboto analyzes the model by slicing it with a cubic grid. The size of the cubes is adaptive and reflects the size of different objects in the scene and their parts. On rare occasions this adaptive mechanism fails, missing some small feature of the model. This can always be cured by reducing the size of the cubes with which Groboto analyzes the model. The Cube Size slider indicates the size of the cubes to be used, as a fraction of their normal size.

If the model meshes OK with standard cube size, using smaller cubes will not change the resulting mesh at all. This is *not* a mesh quality setting. Using smaller cubes slows down mesh generation and increases memory use (very significantly for complicated models and really low Cube Size settings). This feature should only be used when the default setting 1.0 produces an incomplete mesh or no mesh at all.

OBJ Export Settings & Tools

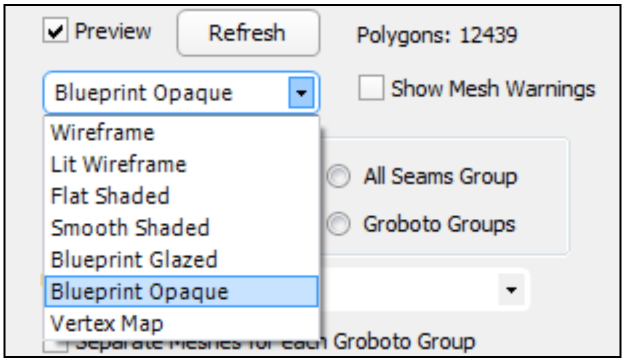
Mesh Export and Preview



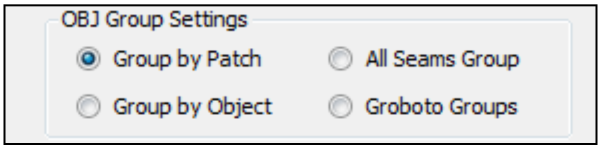
To export unified mesh with the current settings as OBJ file, click the Export to OBJ button. Typically you'd want to look at the mesh before exporting. To do this click the Preview checkbox or Refresh button. Preview checkbox indicates a display mode: when it's off Groboto shows its normal (non-polygonal) display of the model. Only in this mode can you perform most operations in Groboto - add and modify objects, edit boolean clusters, etc. When you switch Preview on, Groboto converts the current scene to unified mesh and displays the mesh. In this mode you cannot do any editing of the model. To continue editing, switch Preview off (Preview can be toggled on/off by pressing M).

What you can do while Preview is on is use the camera controls to navigate around the mesh, change mesh control settings in this dialog box (and select objects by clicking on the mesh with the Select tool and use object-specific mesh settings in the Selected Object Properties tab on the right) and regenerate the mesh with new settings by clicking the Refresh button.

While viewing the mesh you can use several OpenGL display modes available in the display mode pop-up:



OBJ Group Settings


OBJ Group Settings
● Group by Patch
○ All Seams Group
○ Group by Object
○ Groboto Groups

OBJ files support grouping of mesh faces into groups (should not be confused with Groboto groups, which are groups of objects). Since Groboto creates meshes from collections of objects, it is very useful to group faces according to which object they came from. Actually, you can choose among several different levels of face grouping, in the order of increasing number of faces per group:

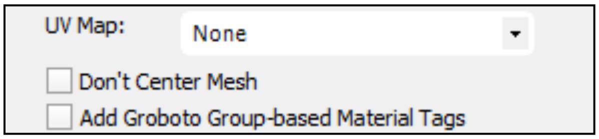
* Group by Patch - a patch is any piece of a surface bounded by its intersection seams with other surfaces. A free-standing sphere is a single patch, a free-standing cylinder is three patches - sidewall and two caps. When objects intersect each other you get more patches and they become smaller. With this option faces from every patch form a separate group in the OBJ file. A patch is always smaller than an object.

* Group by Object - faces from all the patches of any Groboto primitive are grouped together, even if those patches are disconnected as the result of intersection (e.g. a thin cylinder passing throw the center of a sphere - its disconnected ends will be in the same group).

* Groboto Groups - faces from all the objects belonging to a Groboto named group are put into a single OBJ group.

* All Seams Group - divides the entire mesh into just two groups of faces: all quads that run along the seams (seam strips), and everything else. This is a special option, unrelated to the patch-object-Groboto group hierarchy above.

Additional OBJ Settings

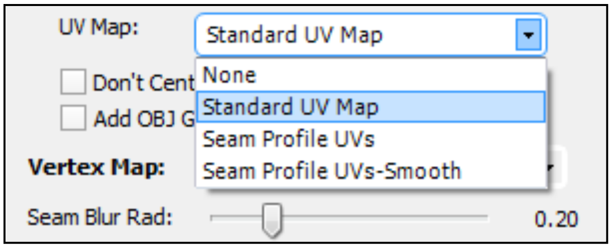

UV Map: None
☐ Don't Center Mesh
☐ Add Groboto Group-based Material Tags

The 'Don't Center Mesh' checkbox: by default, when this checkbox is not checked, Groboto centers the output mesh around the point 0,0,0 (placing it into the averaged center of all the primitives in the scene). There might be cases when you want to export parts of your scene as separate OBJ files (you hide and reveal various groups of objects, and do Export to OBJ for these partially hidden scenes). If you want to reassemble the entire scene in some other program from these separate OBJ files, and you want the parts to maintain the relationship in space they had in Groboto, check the Don't Center Mesh checkbox when exporting.

The 'Add Groboto Group-based Material Tags' checkbox: checking this checkbox will assign a separate OBJ material tag to the part of the mesh generated from each Groboto Group of objects.

UV mapping

When Groboto outputs unified mesh, it can automatically create UV mapping for it. All you need to do is choose Standard UV Map in the UV Map pop-up before clicking Export to OBJ:



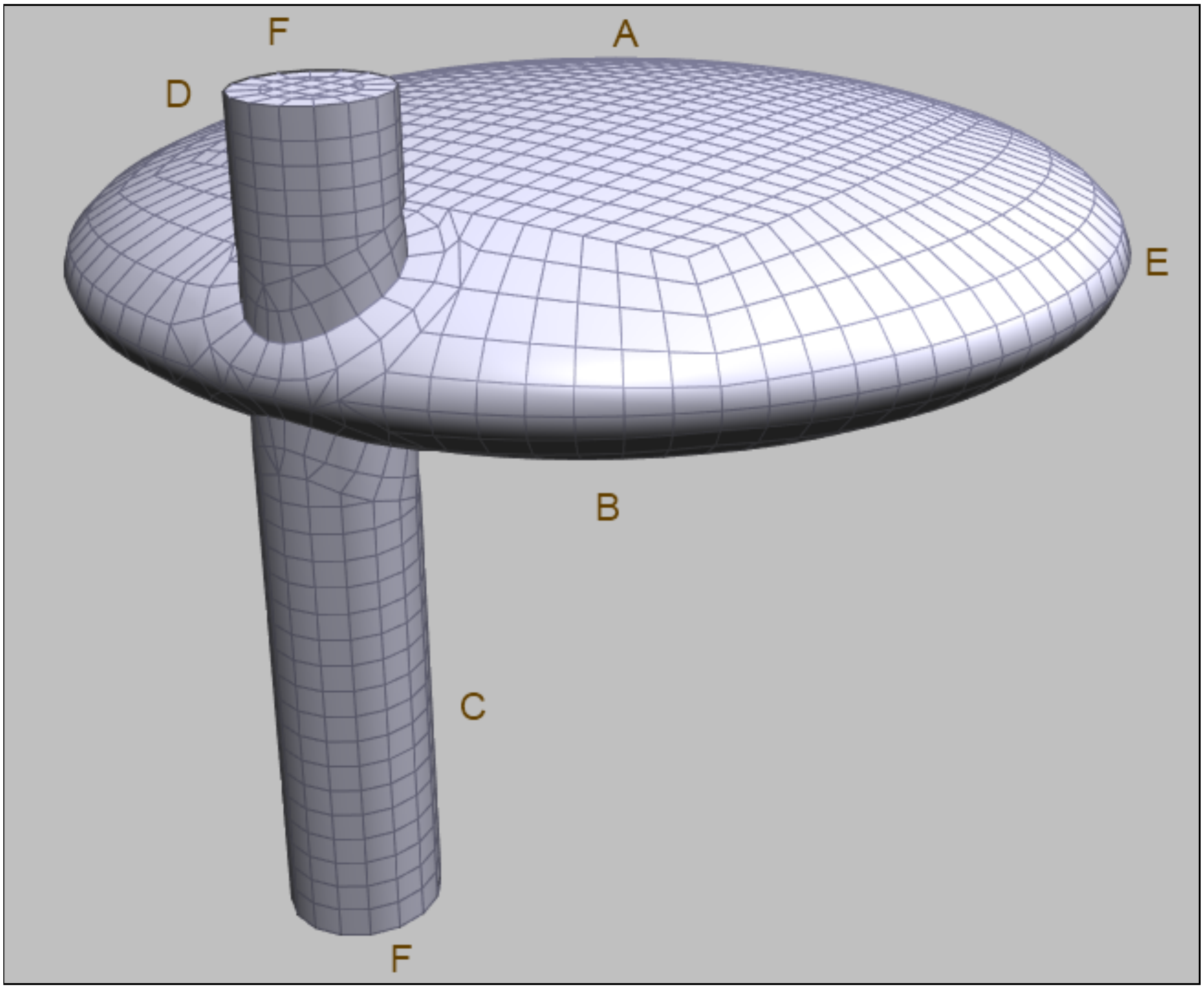
UV Map: Standard UV Map

Seam Blur Rad: 0.20

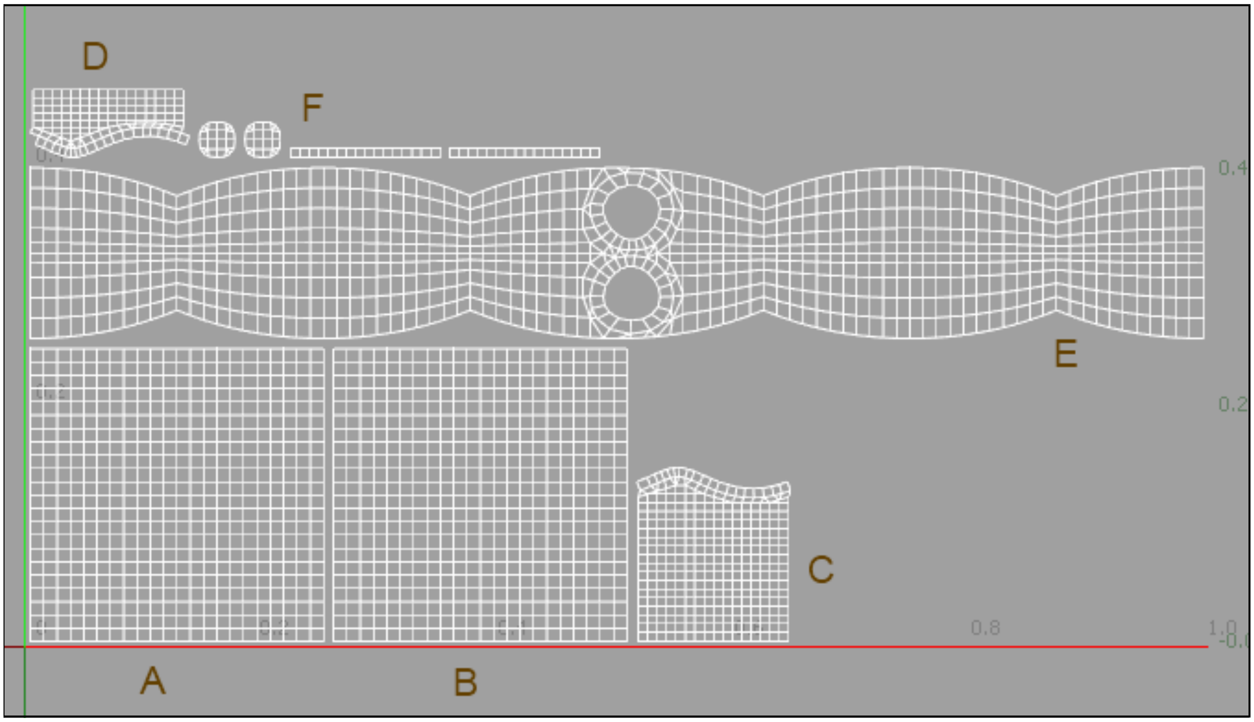
The other two UV output options - Seam Profile UVs, simple and Smooth - generate only V values and only in the seam strips part of the mesh. The purpose of this is not mapping of 2-D images, but generating simple 1-D masks across the seam strip, an older and more limited version of what we do now with Vertex Maps and Bitmaps (see below). This older method is described here: http://www.zbrushcentral.com/showthread.php?100585-Who-Can-it-Be-Now-Challenge&referrerid=74302

Groboto UV mapping is purely object based. Every Groboto model is a collection of primitive objects, and every object (or more precisely, every surface of every object) gets its own separate piece of UV map (or even 2 or 3 pieces – see below). This allows Groboto UV mapping to be fully automatic, there is no need to set up planar, cylindrical or spherical projections, which in any case would most likely be of little value given the topological complexity of most Groboto models. On the negative side, Groboto always UV-maps different surfaces independently, which tends to create too many independent pieces in the UV map.

Let’s look at this simple model (shown as mesh preview in Groboto) as an example of Groboto-generated UV mapping:



What you see below is this model exported as OBJ with Standard UV map enabled, imported into Modo and shown in its UV window:



As was mentioned above (page xxx) each primitive surface in Groboto can have up to three different parts in its mesh, one cylindrical and two polar. Looking at the button ellipsoid portion of our model, we see the large square mesh piece in the middle near the top (A), and there is an identical square mesh piece on the bottom (B). These are the two polar parts; the rest of the mesh is cylindrical in structure, it goes all the way around the equator of the ellipsoid (E). We see these three parts as separate pieces in the UV map.

The three pieces have to be separate UV maps because it is impossible to unwrap a sphere or an ellipsoid onto a UV plane in one piece without creating a severe distortion. On the surface in 3D we have quads all about the same size. In UV some of them will be much larger than others, or will have severely distorted shapes. With our three-piece mapping, every square quad in 3D is reasonably square in UV. This means that if you map an image onto the surface using our UV mapping, it will be reasonably distortion-free. On the other hand, three separate pieces means that you cannot have a single uninterrupted image mapped onto the entire ellipsoid.

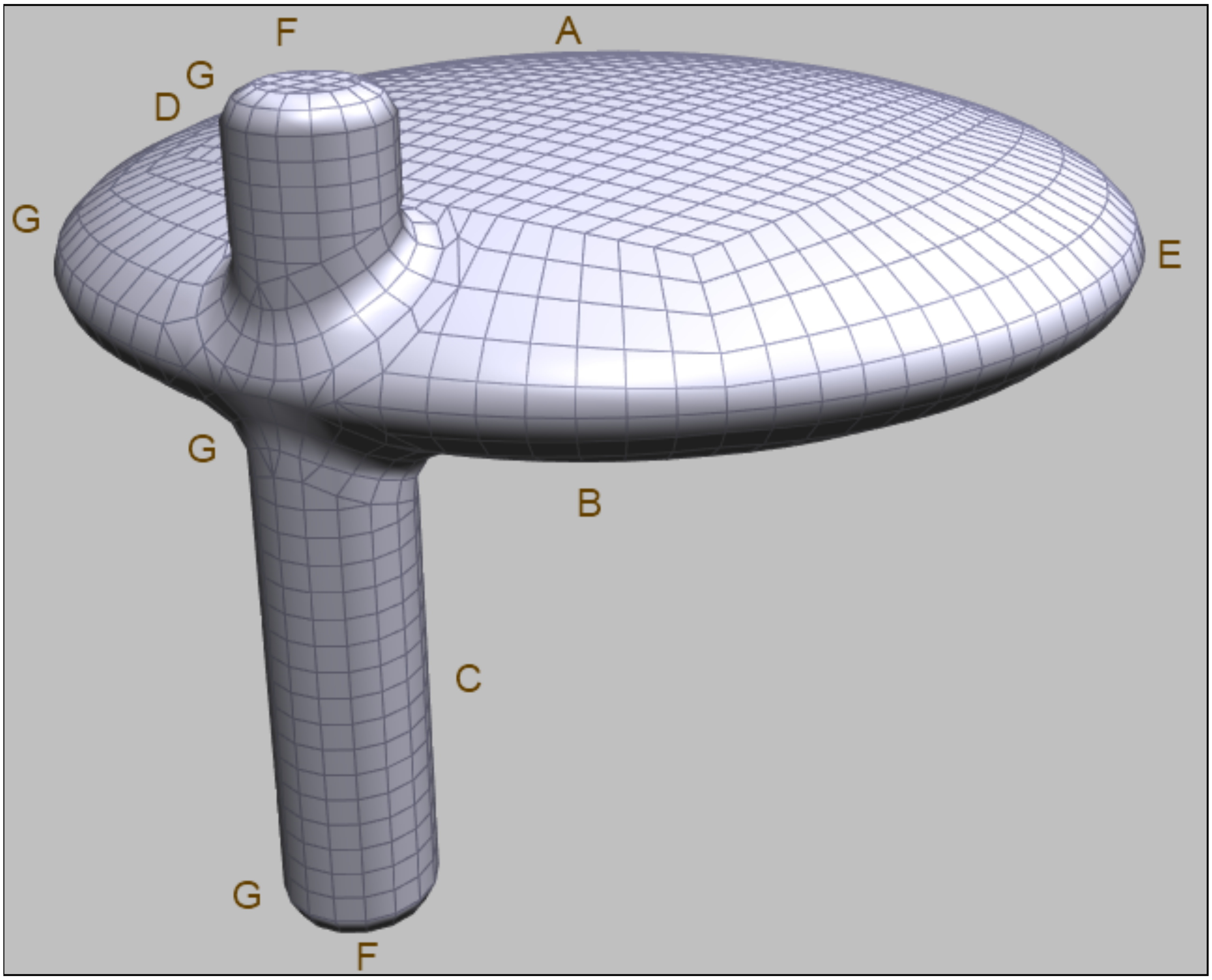
While we cannot avoid mapping a surface into several UV maps on many occasions, we can shift the boundary between different maps if necessary. For instance, you might discover that the boundary between the polar and cylindrical maps on the surface runs right through the spot where you wanted to have some meaningful unbroken image, like text. The boundary could be shifted away from this spot by using the Cap % slider in the Selected Object Properties panel (see Polar and Cylindrical Mesh, page xxx). In the above example, by setting the Cap % value for the button ellipsoid low, we could have made the square maps A and B much smaller, thus covering almost the entire surface of the ellipsoid with a single cylindrical map E.

Note that the cylindrical piece E has two holes in it – this is where the cylinder punctures the surface of the ellipsoid. Near the holes the mesh has a different structure than elsewhere – its rows go around the holes (one row in this case). This is the seam strip, the half of it that lies on the surface of the ellipsoid. Even though it's a very distinct piece of the mesh, it gets UV-mapped together with the rest of the ellipsoid surface. This is what happens when Seam Smoothing is **not** used – when the model has sharp seams like this one. When smoothing is used (see below), seam strips become separate pieces of the UV map.

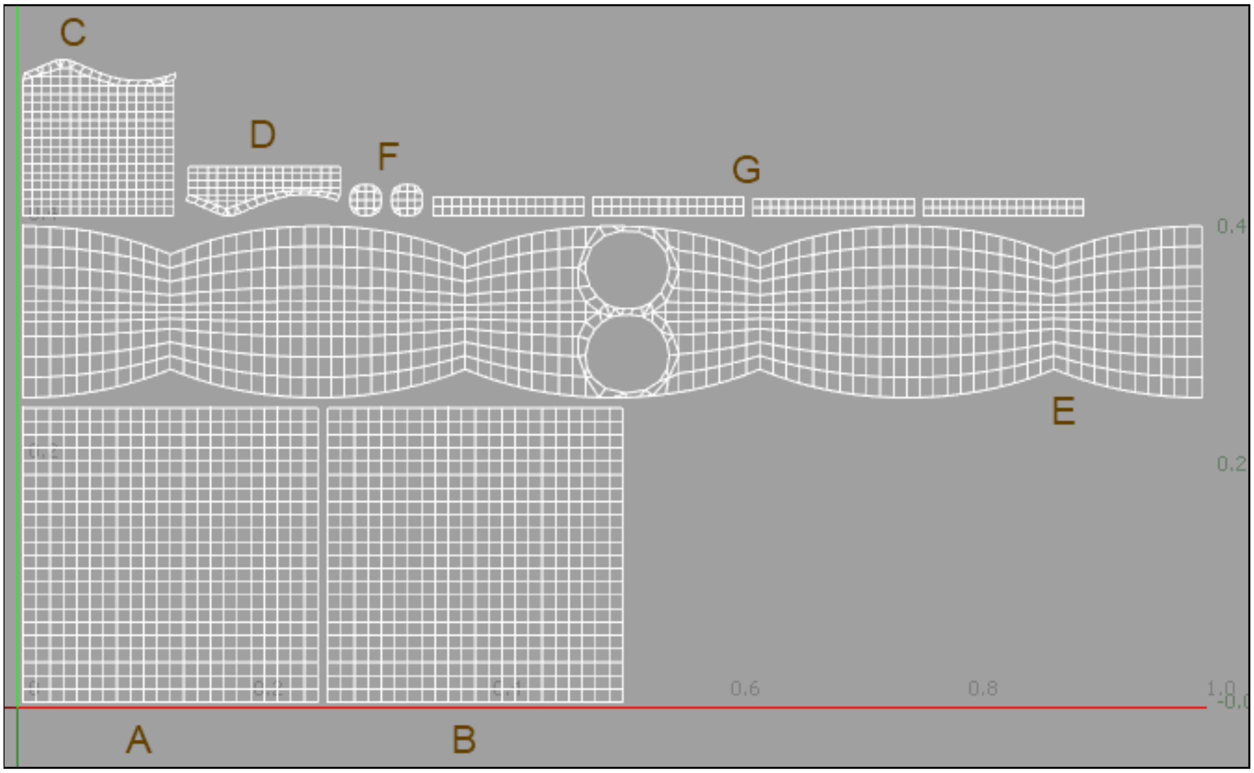
The sidewall of the cylinder appears in UV as two separate pieces, C (lower) and D (upper), because the cylinder is broken by its collision with the ellipsoid. The missing part of the cylinder does not map anywhere in UV because it does not produce any mesh. Note again that while most of the mesh in these two pieces is regular vertical and horizontal grid (native mesh of the cylinder), right on the upper edge of C and the lower edge of D there is a single row running along the curved edge. These are the other halves of the seam strips, again UV-mapped together with the rest of the surface they are attached to, the cylinder.

And finally, we have the two flat caps of the cylinder (F), each consisting of the inner polar piece and the cylindrical band running around the rim. It should be noted that various pieces are arranged in the square UV map based on some global considerations like leaving as little unused space as possible, and that means that pieces of the mesh that are near in 3D could end up far away from each other in the UV square.

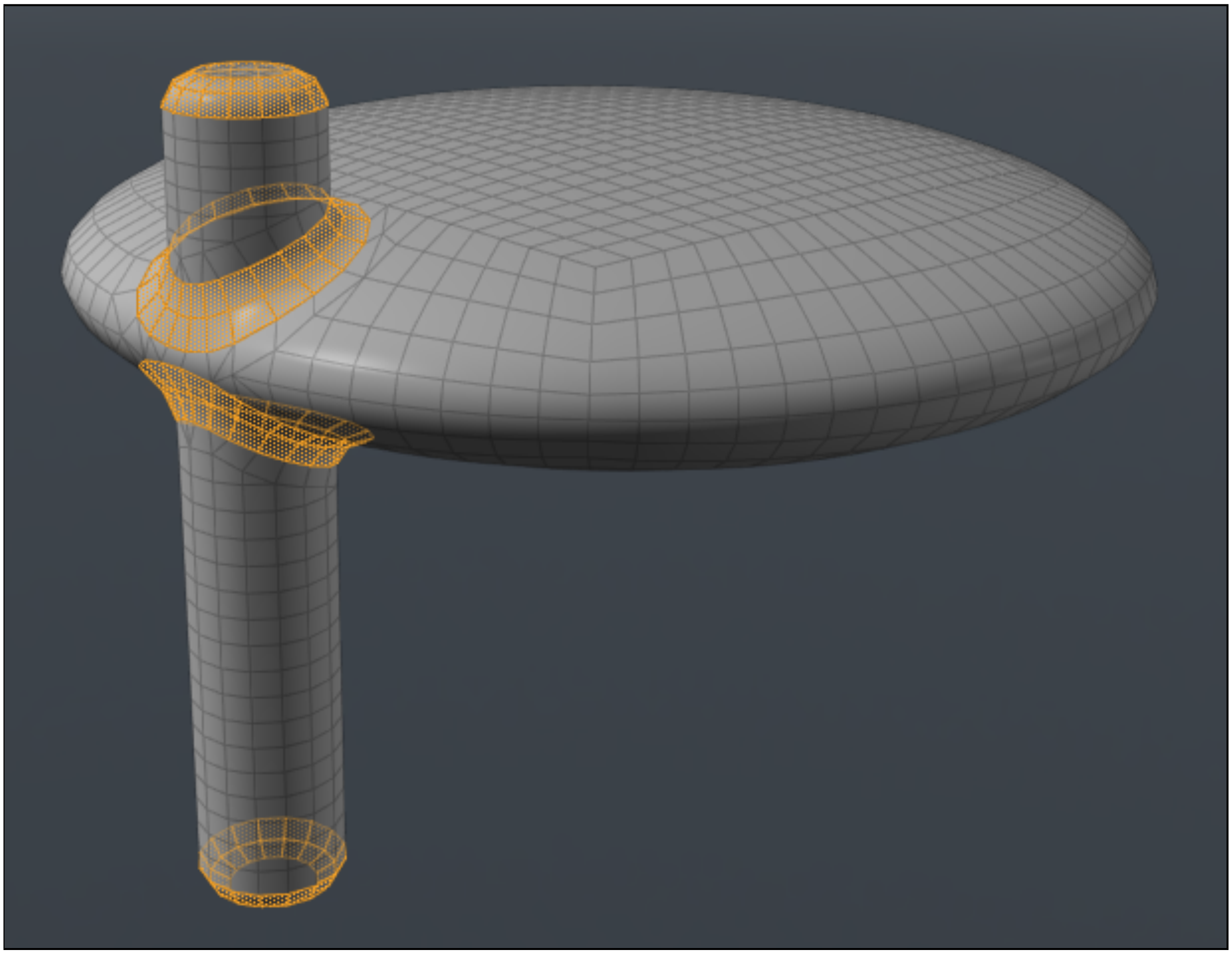
Now let's take the same model, but enable Seam Smoothing (see page 20):



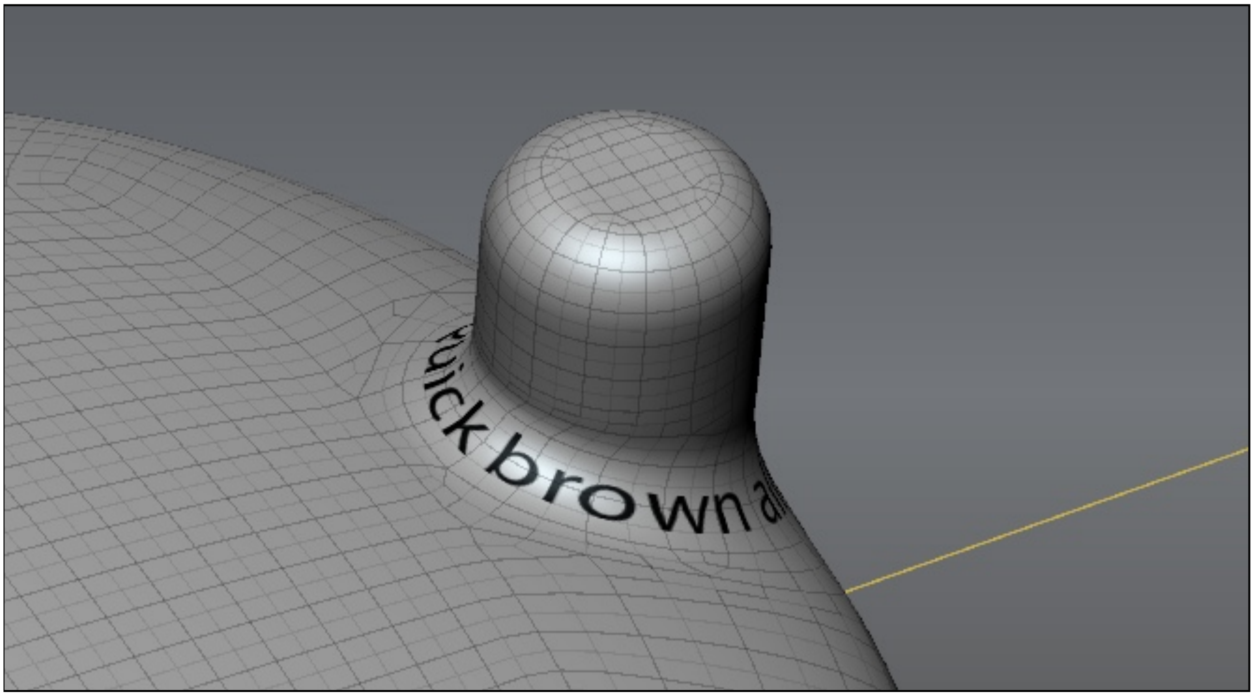
This is the UV map we get in this case:



Everything remains the same except seam strips. The holes in the ellipsoid equatorial band became larger because the half-strips attached to them were removed, as were their corresponding halves attached to the top of C and the bottom of D; these halves were glued together to form complete seam strips, which now appear as separate UV pieces (G). In the image below we see them marked on the mesh in 3D. There are four of them altogether – two strips between the ellipsoid and the cylinder and one between the cylinder and each of its two caps.

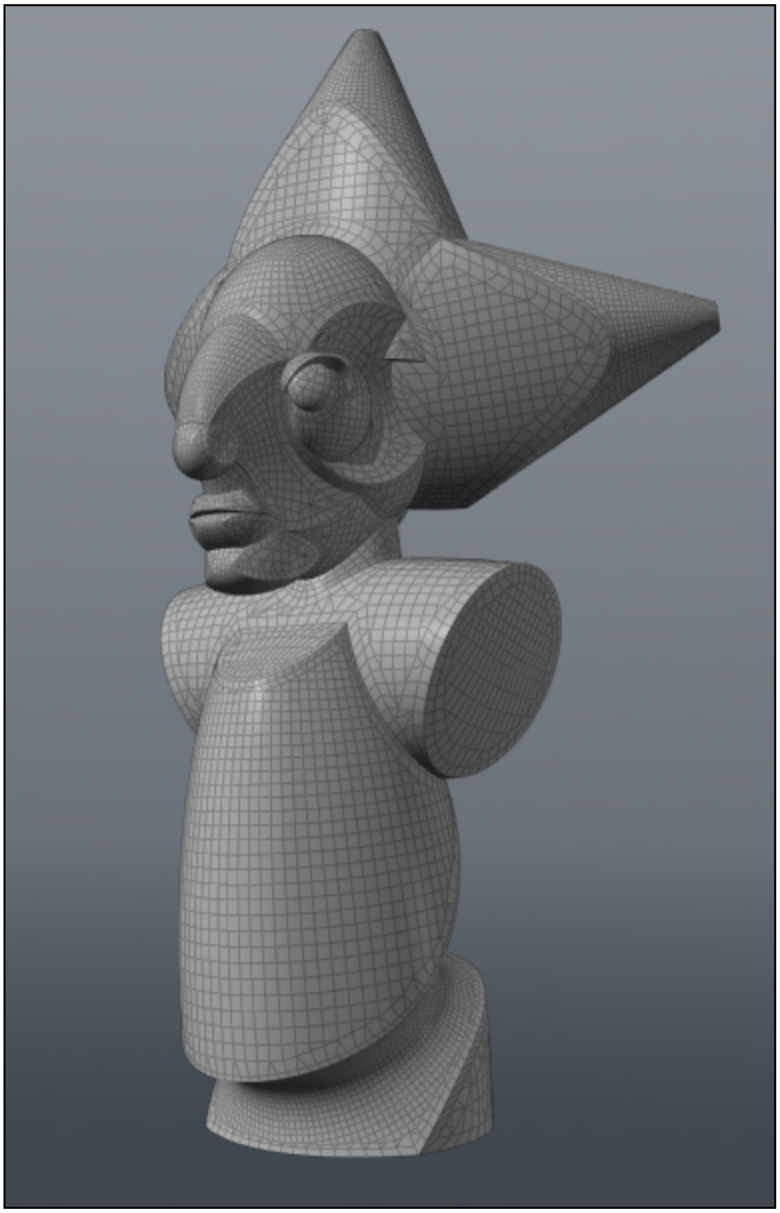


Here UV mapping on the strips is completely independent from that on the ellipsoid or cylinder, and very simple – U runs along the strip, V – across it. This is why in the UV square seam strips always appear as horizontal rectangles, no matter how curved they might be in 3D space. The mapping from UV to 3D is completely distortion-free on the strips; for instance, if we map onto one of these horizontal rectangles G regular horizontal text, this is what we get in 3D:

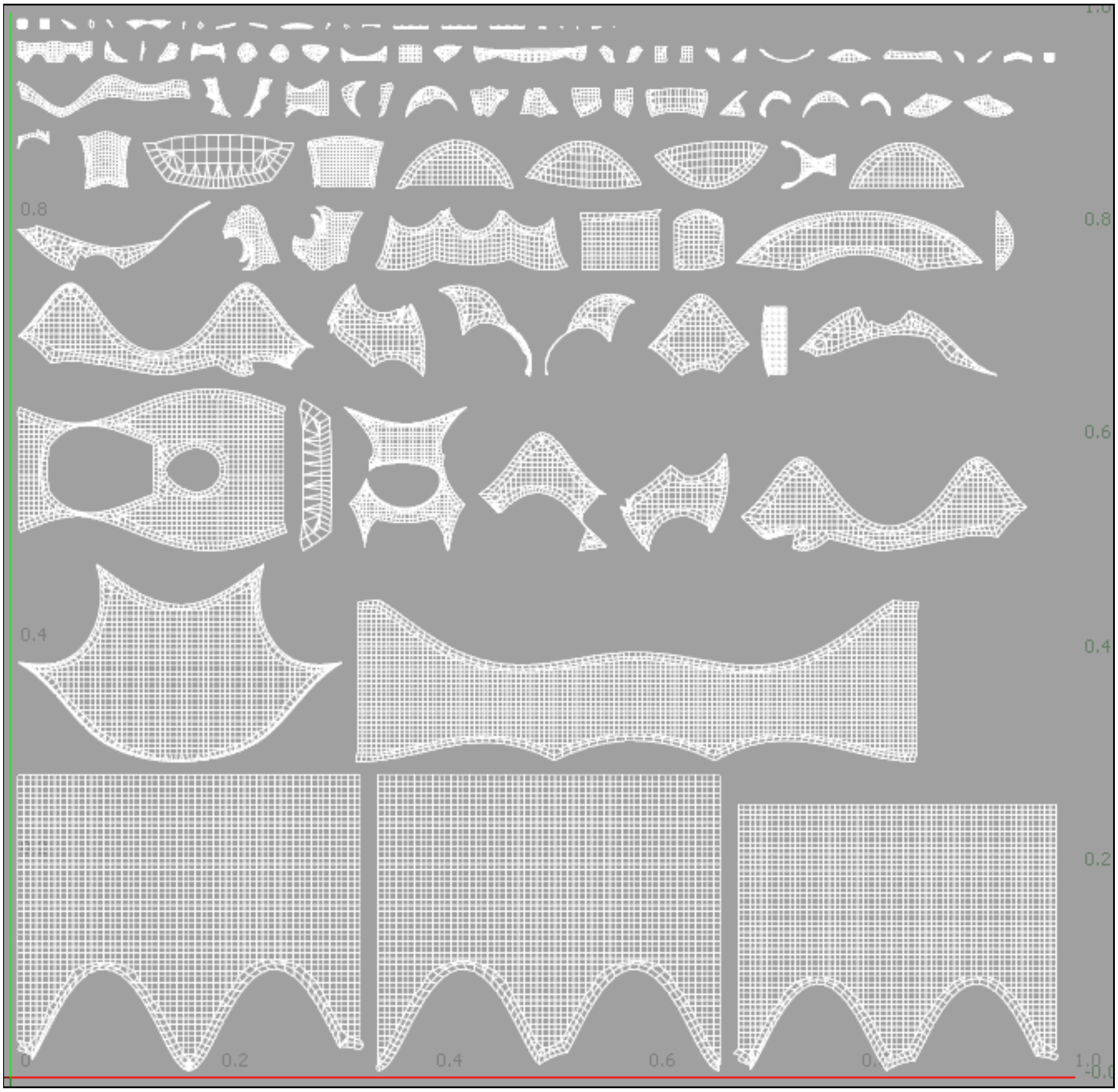


Again, the price we pay for separate distortion-free mapping on the strips is that we cannot have one continuous image covering both the strip and the surface of the ellipsoid or cylinder outside of the strip. Mapping of complicated surfaces onto a plane in one piece is generally impossible without severe distortion or overlaps; Groboto's automatic mapping provides a couple of different flavors of distortion-free mapping in several disconnected pieces.

Here's an example of something a bit more complicated:



And its UV map:



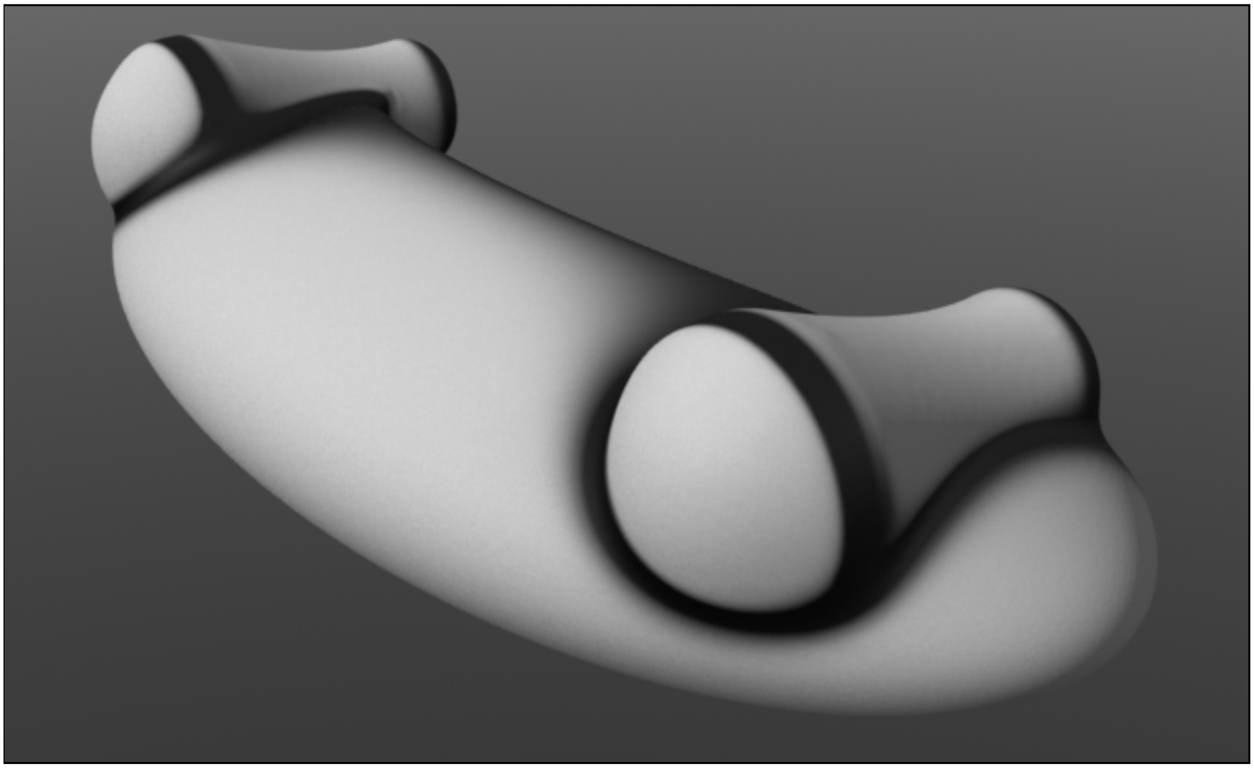
Groboto UV mapping creates UV pieces that are fairly large and always sensible, related to the model's geometry, making it well-suited for both direct 2D painting and the 3D paint tools found in many 3D apps. Exporting your GroBoto model to your 3D paint program with the Standard UV Map option
enabled is all that is required. Groboto maps every point on the model to a unique location in
the UV square, without distortion or overlaps.

Vertex Maps and Bitmaps

This section describes something that could be used for a whole variety of different purposes, from texturing to geometry modification; the only common thread that runs through it all is the idea of creating a function or distribution on the surface of the model that is aware of the structure of the model - seams, patches, etc.

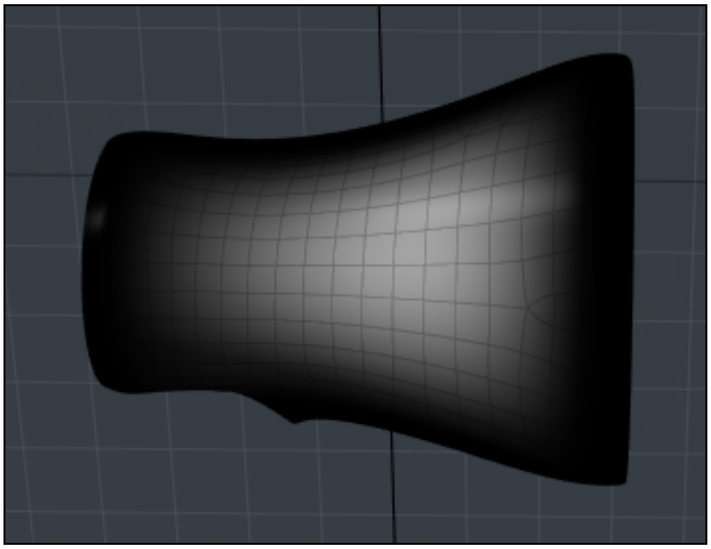
Suppose you want to color the surface of the model so that the seams between patches are darker while the areas of the patches away from the seams are lighter. Or you want to deform the model by blowing out the middles of the patches while keeping the seams immovable. Typically, procedures of this sort are meaningful because they correspond to the basic geometry of the model. Of course, they could be combined with completely free and unstructured painting on the surface, or, conversely, with something as structured as a simple vertical gradient in space, to create a wide variety of texturing and deformation effects. Whatever other tools you are using, adding to the mix effects that are automatically aware of the structure of the model is frequently valuable.

On Groboto-generated models such geometry-aware surface effects often look really beautiful:



But how to produce such a distribution of color, synchronized with geometry? Certainly no predefined color gradient in space is intelligent enough to know where various pieces of this model intersect each other. Nor is it a matter of creating a gradient in some native direction on each primitive surface, like along the axis on a cylinder, or pole to pole on a sphere. Again, how would it know where the cylinder intersects the sphere?

Groboto's answer to this problem is the notion of a *patch field*. A patch in Groboto is any piece of a surfaces bounded by seams, where it intersects with other surfaces. Groboto can generate on any patch a function that is zero everywhere on the boundary of the patch and rises inward toward the center of the patch, reaching the maximum value of 1. Here's an example from the model above:

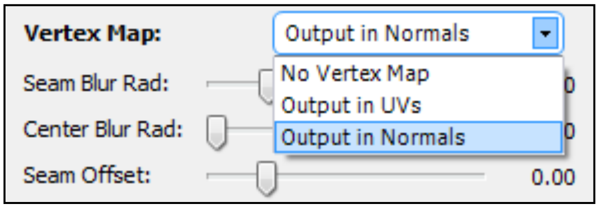


What this function is used for is entirely up to you; in the example above zero means black diffused color, one means white color. It could have been used as displacement, or as vertex weight map to control some deformation tool. Groboto allows you to control the field function in some simple ways, like changing the steepness of the inward gradient and changing how deep into the patch the gradient starts. The field is generated for all the patches; you can decide later how to use it for each patch and whether to use it at all.

There are two delivery methods for the field function: vertex maps and bitmaps. With vertex maps Groboto calculates the value of the field function for every vertex of the mesh it generates, and stores this vertex value in one of the two data sets available in the OBJ format, beside the mesh itself: UVs or surface normals. You can use this approach if the program where you export your Groboto-generated mesh allows you to interpret either of these two OBJ values as vertex weights. The main drawback of this delivery method is its low resolution. Its resolution would normally be sufficient when used as vertex weight map to modulate the effect of some mesh deformation tool, since mesh deformations are applied only to mesh vertices anyway.

In all the cases where higher resolution is needed you can output the field as a bitmap. This, of course, also requires using the Standard UV map option, since a bitmap is simply a flat square image and something has to map every point on the surface of the model into this square image. Bitmaps offer tremendous possibilities for hi-res output; Groboto can generate field bitmaps of up to 1 billion pixels (32K squared). One use of fields where resolution is most in demand is for surface displacement.

When outputting vertex maps Groboto creates a single OBJ file as usual. What sort of vertex map data is put into the file is controlled by the Vertex Map pop-up:



Vertex Map: Output in Normals

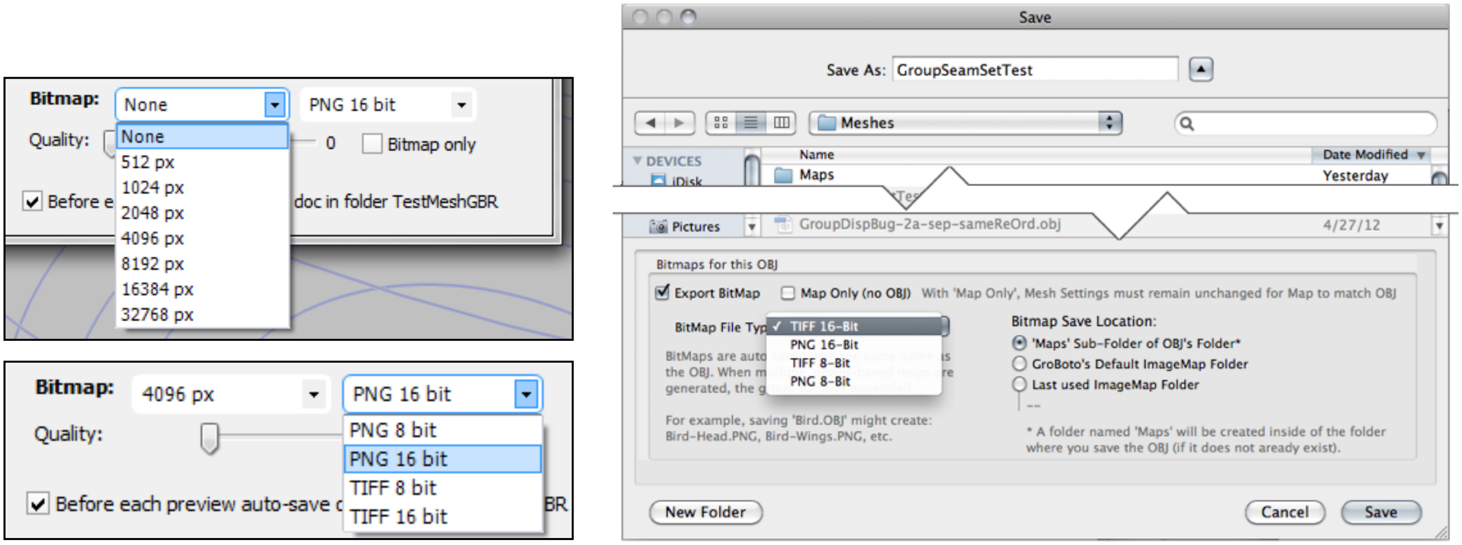
Seam Offset: 0.00

By default there is no vertex map data. If you choose Output in UVs Groboto will calculate the field and store it in the UV vertex values of the OBJ file. Specifically, U will be set to zero, V to the field value (0 to 1). With this option you lose the ability to output regular UVs for texture mapping (the UV output pop-up is disabled when this option is used). To use the field as a mask or vertex weight in the program where you export the model, you typically create a simple square image of a vertical gradient from black to white and apply it as a texture, using the UVs that came from Groboto in the OBJ file (vertical gradient because V is the vertical direction in the image). This maps an image onto the surface whose grayscale value at each vertex equals the field value generated by Groboto for this vertex. All you need to do is to set the mask to this texture. In ZBrush, for instance, it's called Mask by Intensity.

If you choose Output in Normals, Groboto will store the field value in the X component of the surface normal for each vertex (Y and Z components are set to zero). You lose the ability to export actual surface normal data. This is usually not a problem - vertex map field export is typically used to control mesh deformation tools, and if you deform the mesh, Groboto-generated surface normals will have to be thrown away in any case. There is no natural mechanism (like with UVs above) for converting the X component of the surface normal into vertex weight. Modo, for instance, supports it simply because in Modo you can convert any vertex-related data into any other vertex-related data. You can download from groboto.com/v3/Samples the Groboto-to-Modo ToolKit which performs this data conversion automatically.

You can also output the field as a bitmap. The vertex map and bitmap options are completely independent, either one can be turned on or off, and both can be used at the same time. Bitmap output is controlled by the Bitmap pop-up:

You either choose None (default) or you choose the size of the square image to be created, from 512x512 to 32768x32768. With this option turned on, when you click Export to OBJ Groboto will create one or more image files in addition to the OBJ file. The format of the image file is chosen in the file format pop-up (16-bit formats are almost always necessary; 8-bit is too crude in most cases):



With 'Map Only', Mesh Settings must remain unchanged for Map to match OBJ

For example, saving 'Bird.OBJ' might create: Bird-Head.PNG, Bird-Wings.PNG, etc.

* A folder named 'Maps' will be created inside of the folder where you save the OBJ (if it does not aready exist).

Left:Windows Right:Mac OSX on Mac, Sizes are in Output Panel, File format popup is in Save dialog box

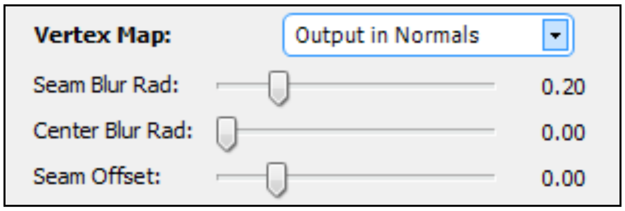
There will be a single image file if you generate a single unified mesh. If you have multiple groups in your scene and use the Separate Meshes for each Groboto Group option (see page 68), each separate mesh will get its own image file. Note that field bitmap size, as well as most mesh control parameters (density, seam width, etc.) could be set separately for each separate mesh (see page 61 below). You don't specify the image filename - it is automatically generated from the name of the OBJ file; in the case of multiple groups each image gets the group name appended to its filename. You also don't specify manually the directory of the image file(s) - Groboto always puts them into the folder 'Maps' (creating it if necessary) inside the folder where you save the OBJ file.

If you check the 'Bitmap only' checkbox Groboto will create the bitmap file(s), but will not create the OBJ file. This is useful in those cases when you want to create several different versions of bitmap file(s) for the same mesh (OBJ). For instance, if you want a bitmap for diffused color and a bitmap for displacement, it's very likely that you'd want different control settings (Blur, Offset - see below) for these bitmaps. You can output the first set of bitmap files when creating the OBJ itself ('Bitmap only' checkbox unchecked). Then you change the bitmap control settings and click Export to OBJ again, this time with 'Bitmap only' checkbox checked. You need to specify a different filename in the Save dialog box the second time, or else the first bitmap file(s) will be overwritten. For consistency, the filename you specify is still of the OBJ file (even though no OBJ file is created); the image filename(s) are generated automatically from the OBJ filename. Do not change any output options other than the map Blur/Offset sliders and bitmap size and quality - the mesh itself has to remain exactly the same as the first time.

The Quality slider is important - using very high quality settings with very large bitmap sizes like 16K can make field bitmap calculations *extremely* long. Lower quality settings reduce calculation times dramatically. Generally the quality is pretty good even with lowest settings. What is acceptable depends on what you use this data for - surface displacement, bump, diffuse color, etc. It's always a good idea (certainly when using large sizes) to start with lowest quality setting (0) and redo the output with higher settings only if necessary.

Patch Fields

Patch field function is controlled by these three sliders in the Unified Mesh output dialog:



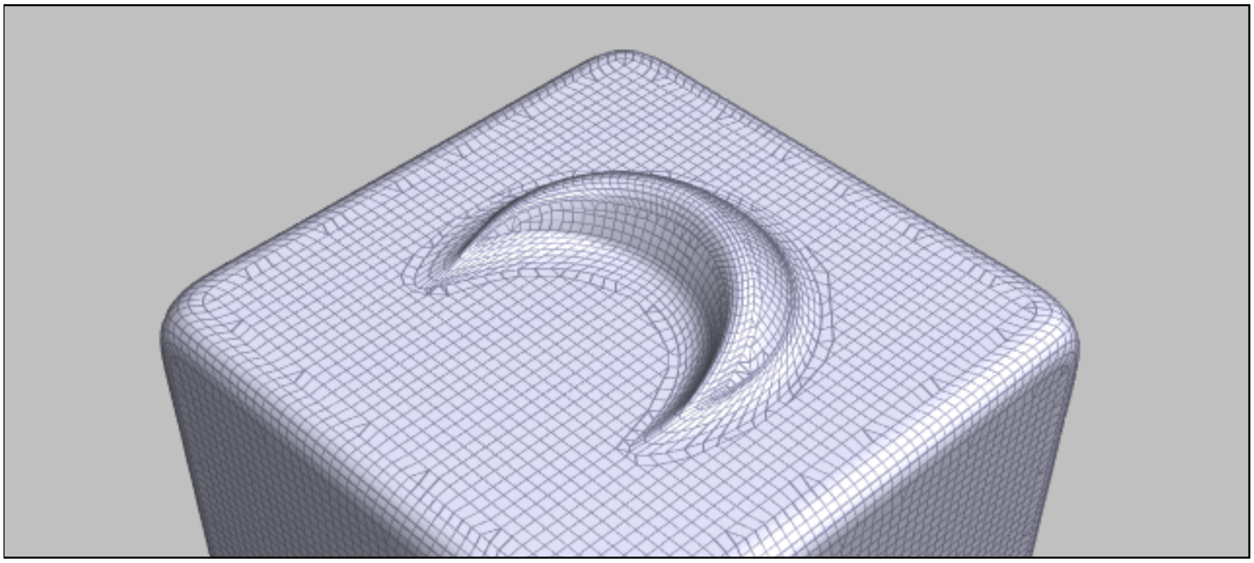
Vertex Map: Output in Normals

Seam Blur Rad: 0.20

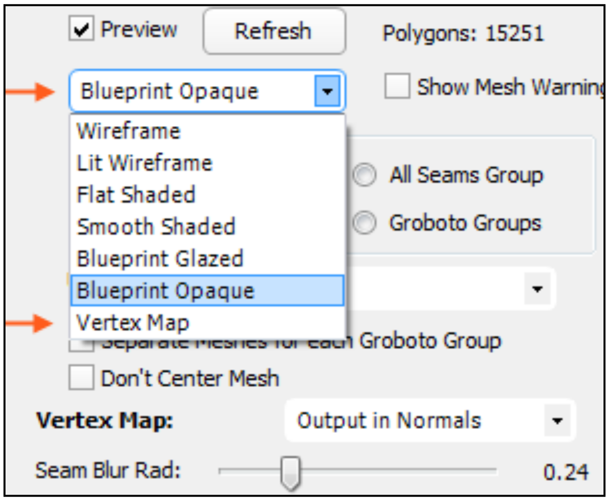
Center Blur Rad: 0.00

Seam Offset: 0.00

The function is the same whether it is output as vertex map or bitmap. The fastest way to try different slider settings and see the results is to use mesh preview with vertex map output. Choose either Output in UVs or Output in Normals in the Vertex Map pop-up (this choice matters only when you hit Export to OBJ; for Preview it makes no difference). To illustrate various Vertex Weight options we'll use this simple model, a crescent embedded into a cube:



The mesh itself, shown above in the Blueprint Opaque mode, remains the same as we try different settings. To see the field distribution on the surface we have a special display mode, available in the display mode pop-up, called Vertex Map:

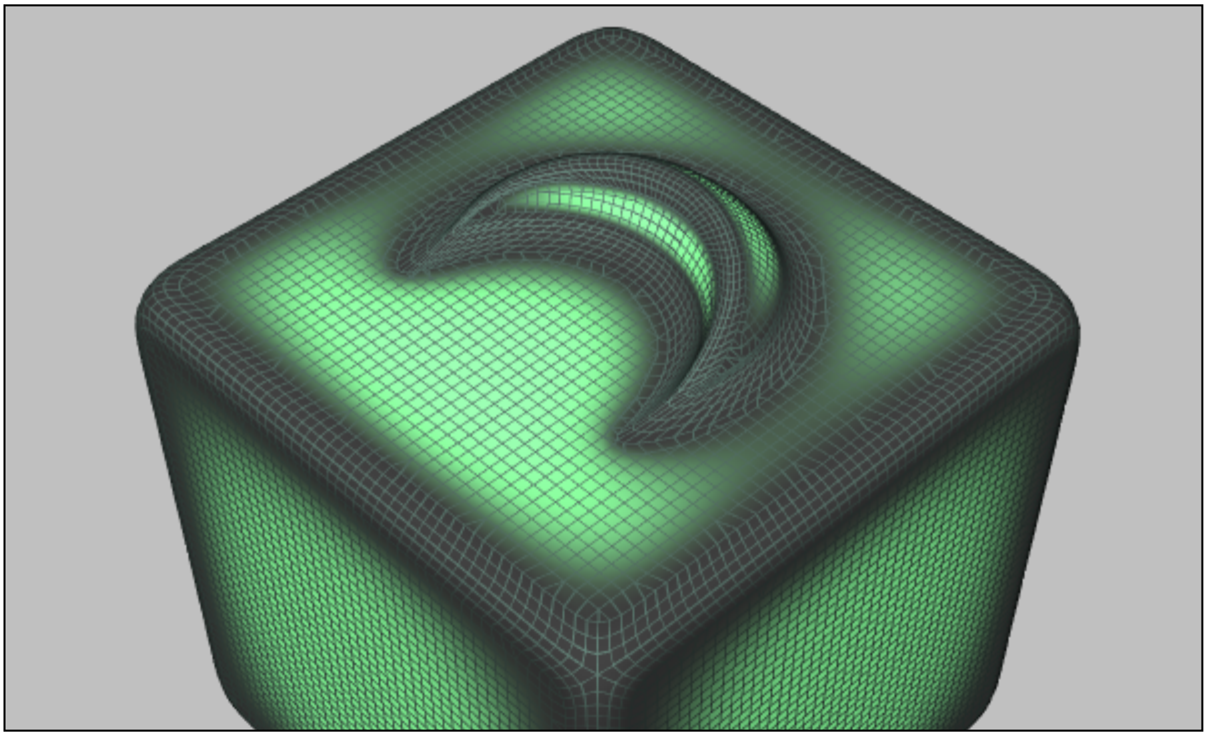


Polygons: 15251

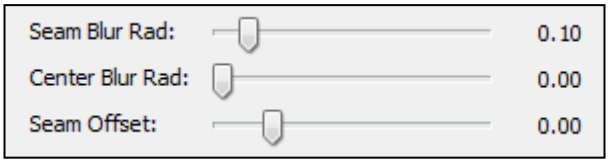
Vertex Map: Output in Normals

Seam Blur Rad: 0.24

Choosing this mode we get the following display:

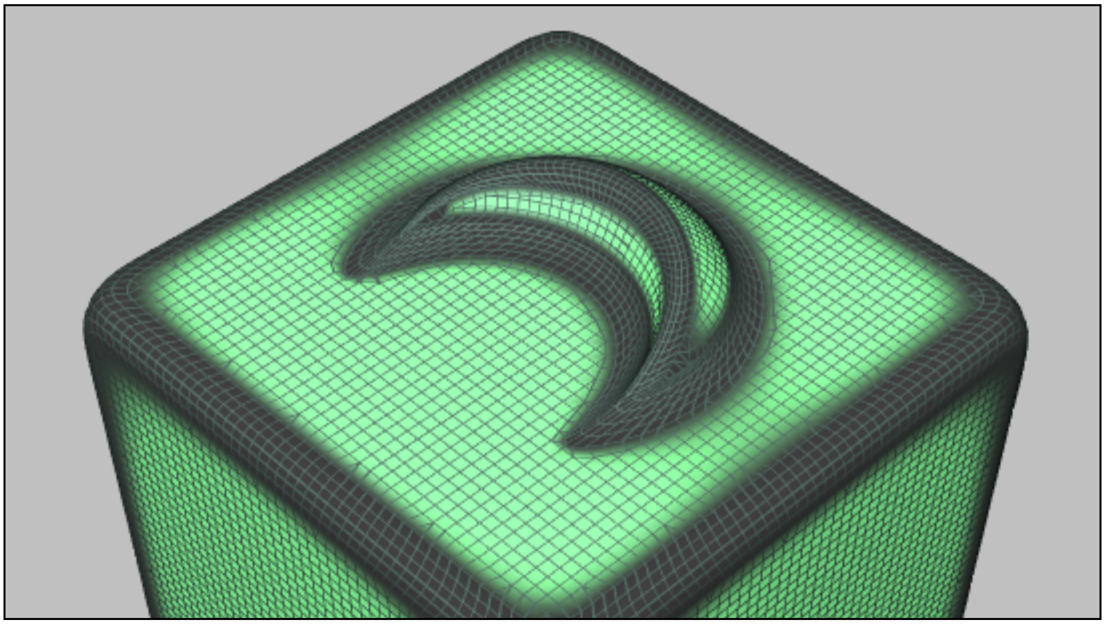


The field distribution is shown as color on the surface, the darkest gray corresponding to 0, the brightest green to 1. The distribution always respects the shape of each patch, falling off towards its boundaries. Now we can study the effect of various slider settings on this distribution. First, let's try this: A small value for Seam Blur radius, zero for others.

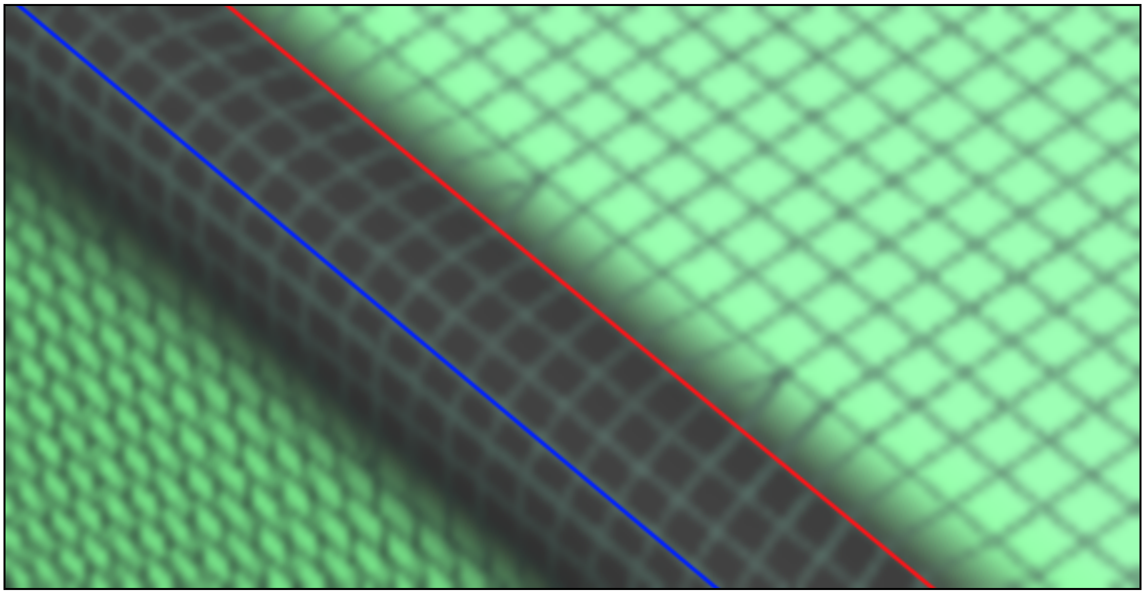


Seam Blur Rad: 0.10
Center Blur Rad: 0.00
Seam Offset: 0.00

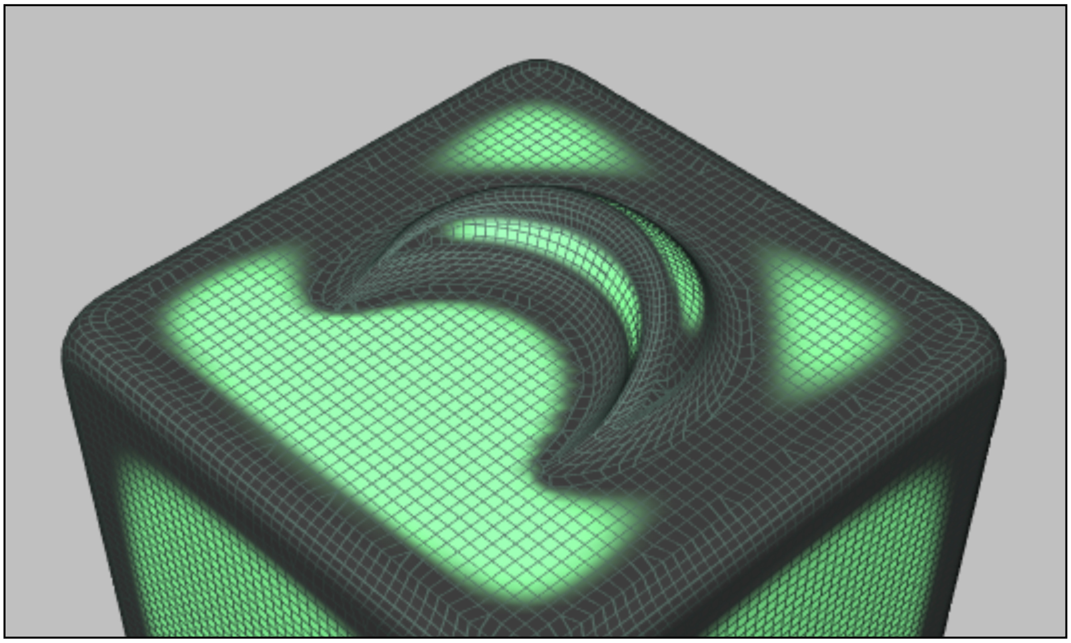
This is the result:



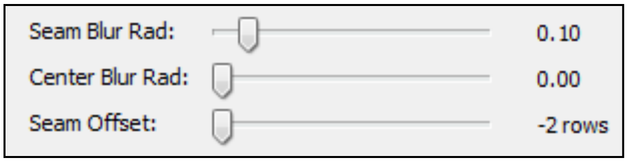
Let's get a closer look at what happens near the patch boundary:



This mesh was generated with three Seam Strip rows. The line marked in blue is the middle of the seam strip, the one marked in red is the edge of the seam strip. The vertex weights are all zero in the strip; beyond the strip they rise quickly to bright green (value = 1). Small Seam Blur Radius means quick rise of the field value. With greater Seam Blur Radius you get a more gradual rise of the field value. The bottom slider, Seam Offset, controls where the rise begins. The default value 0 means the edge of the seam strip, as seen above. Here's what we get if we set Seam Offset = 0.2:

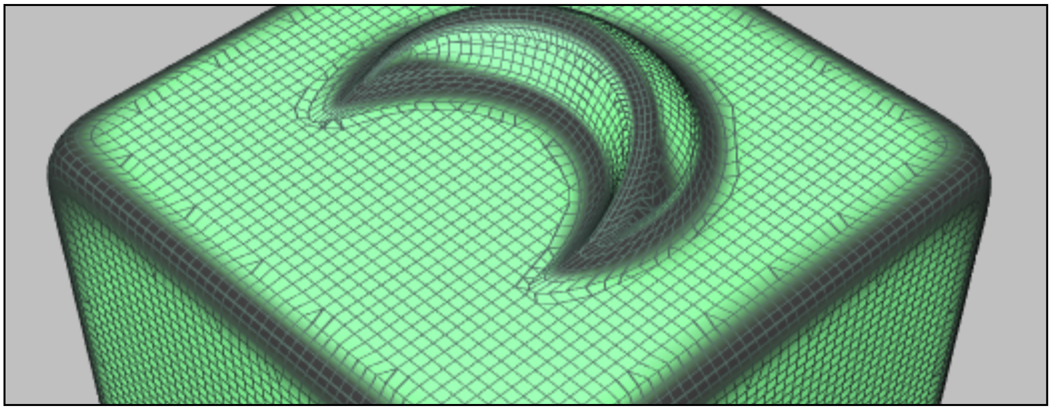


The rise is as sharp as before, but it starts deeper into the patch, some distance from the edge of the seam strip. You can also move the Seam Offset slider to the left from its default position:

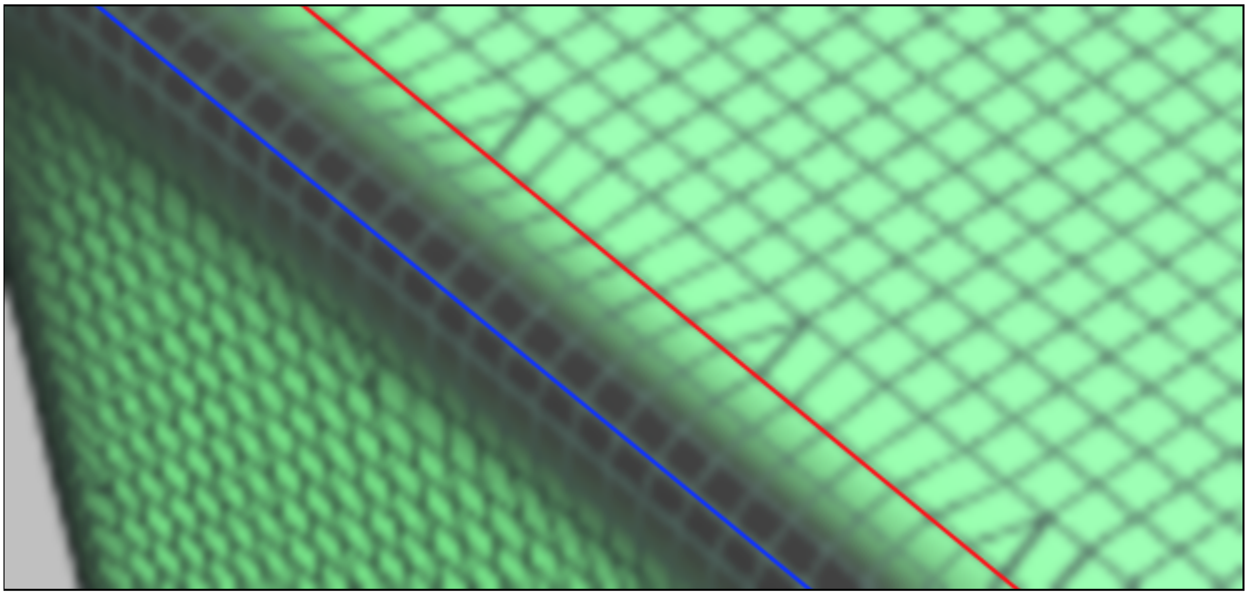


Seam Blur Rad: 0.10
Center Blur Rad: 0.00
Seam Offset: -2 rows

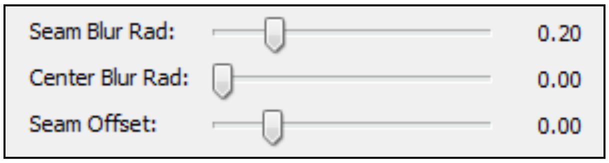
with this result:



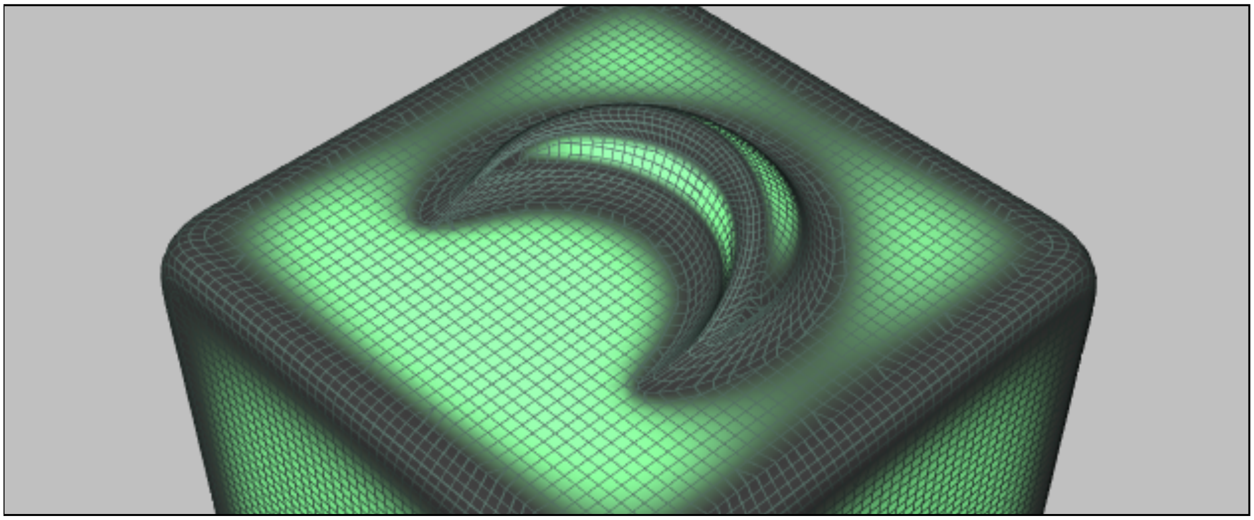
in this case the rise begins in the seam strip, two rows deep inside it, leaving only one row out of three completely dark:



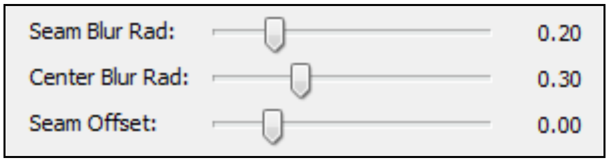
When using negative settings for this slider, to the left of the default zero, we just show the number of rows the field is getting into the strip. On the positive side we show a fractional value from 0 to 1 because outside of the seam strip the boundary of the field is generally not running in the same direction as mesh rows; all we can indicate is a value from 0 (seam strip boundary) to 1 (all the way to the center of the patch). Now let's set Seam Offset back to zero, but use a wider Seam Blur Radius, 0.20:



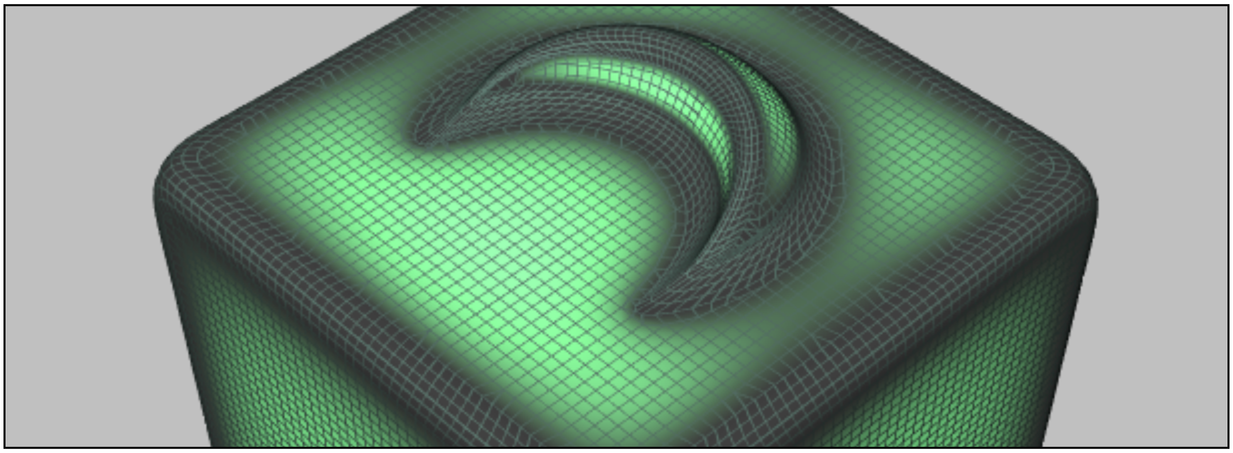
Seam Blur Rad: 0.20
Center Blur Rad: 0.00
Seam Offset: 0.00



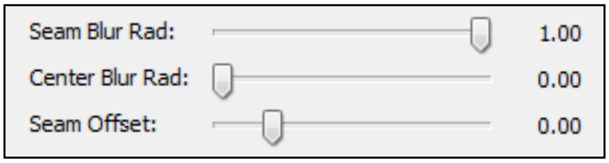
As mentioned above, with greater Seam Blur Radius you get a more gradual transition from dark to light along the boundaries. And with greater values still we can have a really slow gradual transition - with about the same rate of rise in the entire range from dark to light. What the middle slider, Center Blur Radius, allows us to do is to make the transition slower in the brighter areas (towards the middle of the patch), while still keeping the rise pretty steep in the dark areas near the boundary. Here's the effect with Center Blur Radius set to 0.30:



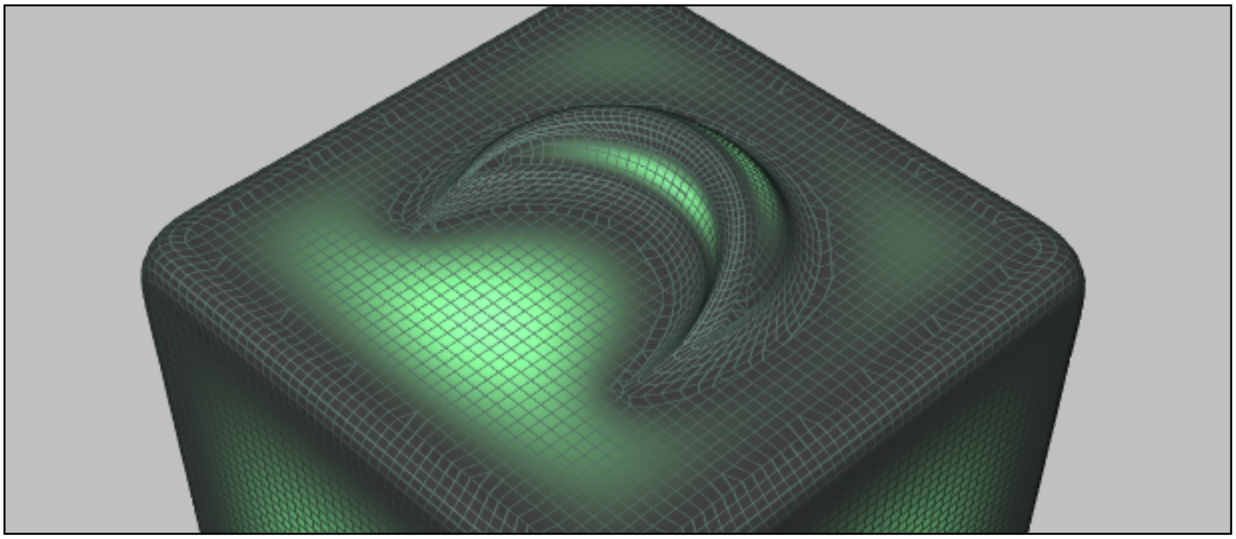
Seam Blur Rad: 0.20
Center Blur Rad: 0.30
Seam Offset: 0.00



The transition is much less sharp now in the bright green area, while near the seam there's not much change. Compare this to very high setting for Seam Blur Radius:

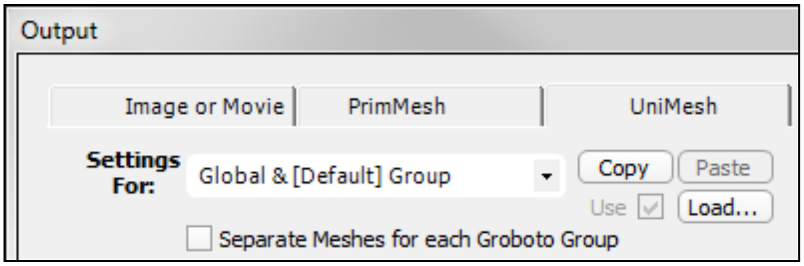


Seam Blur Rad: 1.00
Center Blur Rad: 0.00
Seam Offset: 0.00



where the transition is slow all the way from dark to bright. At higher settings of the top slider the second slider has almost no effect, since the transition is already slow everywhere.

Exporting Multiple Meshes



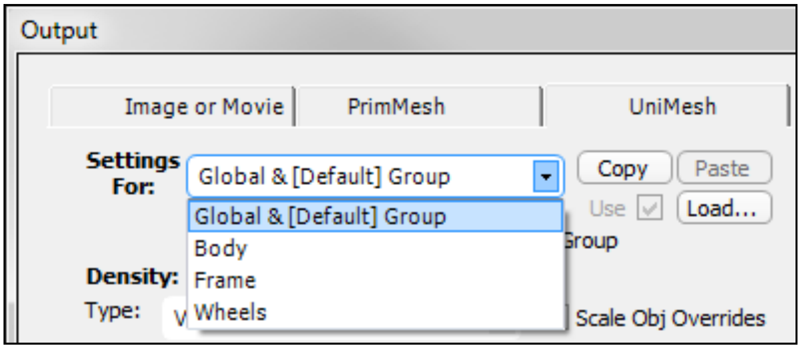
By default everything that you have in your Groboto scene will be converted into a single continuous unified mesh. Of course, if you position some objects or collections of objects in your scene so that they don't touch other objects, they will form a separate disconnected piece of the mesh. But anything that overlaps melds together.

This default behavior can be modified if you use Groboto groups. If you check the Separate Meshes for each Groboto Group checkbox, each Groboto group will produce its own disconnected mesh, even if it happens to overlap in space with other groups. The meshes will simply go through each other.

When using this option, almost all the output mesh settings that you see in this panel can be specified separately for each separate mesh. There are a few settings that are common to all separate meshes:

1. OBJ Group Settings - Group by Patch, Group by Object, etc.
2. Choice of UV Map - None, Standard, etc.
3. Choice of Vertex Map - None, Output in Normals, etc.
4. Bitmap file type - PNG, TIFF and bits per pixel.
5. Bitmap only switch.

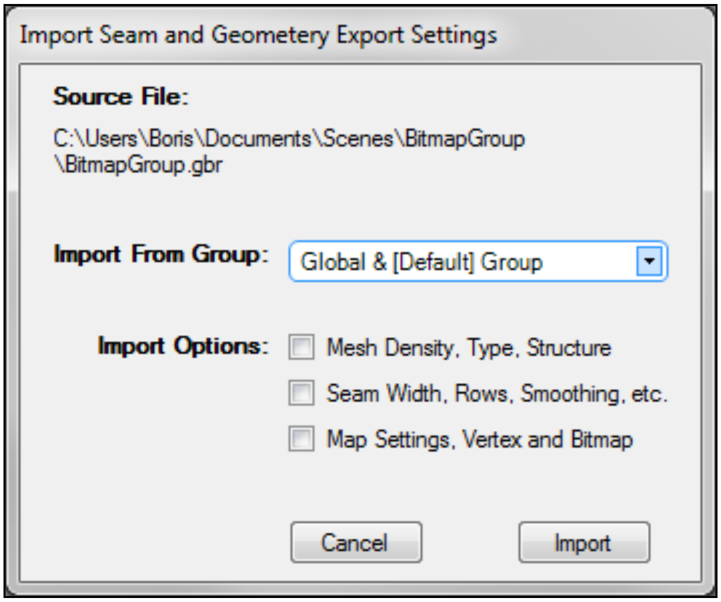
Everything else can be set separately for each separate mesh (uncheck the 'Use' checkbox if you don't want to use separate settings - settings from Global & [Default] Group will be used). The 'Settings For' pop-up in the upper left corner contains the list of groups in the scene:



When you specify settings in the Output panel, you specify them for the group selected in this pop-up (except the common settings listed above). Selecting a different group in the pop-up loads its settings into the panel. If you uncheck the Use checkbox to the right of the pop-up, this group will use the Global [Default] settings; its own separate settings will remain undisturbed, for possible use in the future. Settings can be copied from one group to another by choosing different groups in the pop-up and clicking the Copy and Paste buttons. To create Groboto groups and place objects into groups use the Selected Object Properties (leftmost) tab of the Main panel to the right of Groboto workspace.

Load mesh settings

Mesh settings can also be imported from another Groboto file. Unlike Copy and Paste above, this is not related specifically to files with multiple mesh settings. If you have multiple group settings, select the group you want to import settings into in the Settings For pop-up (see above) before you click the Load... button. Clicking the Load... button brings up the file selector where you choose the Groboto file from which you want to import the mesh settings (only mesh settings will be read from this file, nothing else). Next the following dialog box appears:



Import Seam and Geometery Export Settings

Source File:
C:\Users\Boris\Documents\Scenes\BitmapGroup\BitmapGroup.gbr

Import From Group: Global & [Default] Group

Import Options:
- ☐ Mesh Density, Type, Structure
- ☐ Seam Width, Rows, Smoothing, etc.
- ☐ Map Settings, Vertex and Bitmap

The Import From Group pop-up has the same function as the one in the Output panel - you can choose from which group in the source file the mesh settings will be imported (if it has several). The checkboxes below allow you to choose a particular subset of mesh settings for import (what is not checked will not be modified):

Mesh Density, Type, Structure - everything on the left side of the Output dialog above the Seam Strip section.

Seam Width, Rows, Smoothing - the lower part of the left side, beginning with the Seam Strip section.

Map Settings, Vertex and Bitmap - the lower right side, Vertex Map/Bitmap control sliders and Bitmap sizes and quality settings.