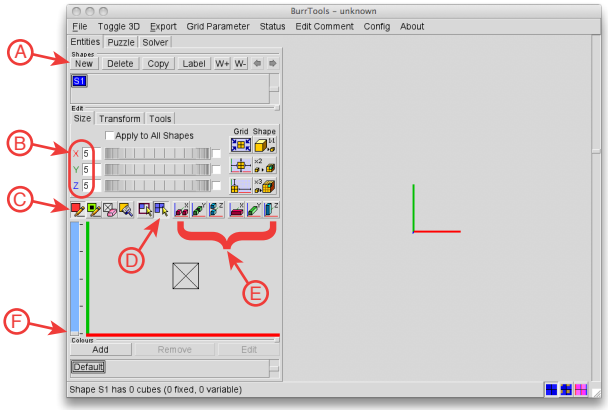
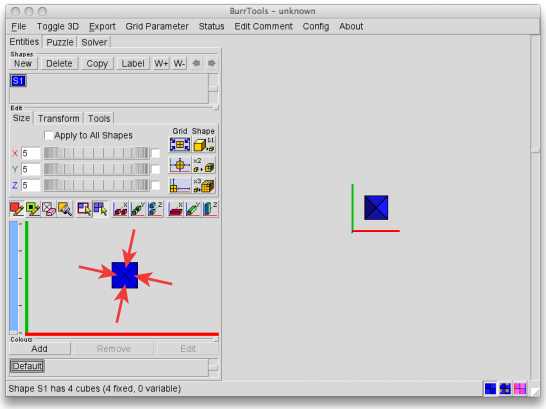
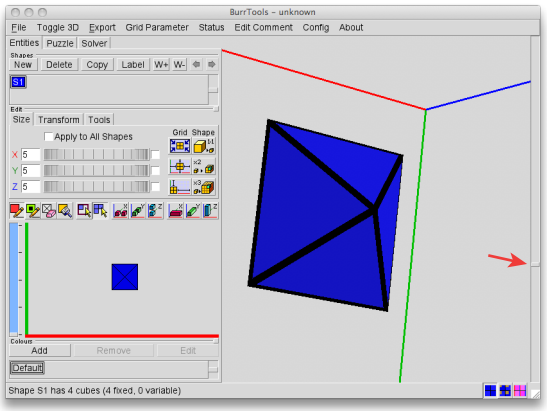
Step 1: Open the File Menu and create a “New” Document using the “Rhombic Tetrahedral” space grid.

A) Create a new Shape.
B) Set XYZ size to 5 units each.
C) Enable the “fixed pixel” drawing tool.
D) Enable the “painting” cursor.
E) Disable all “mirroring” and “layer spanning” tools.
F) Set the grid editor to the back (lowest) layer.

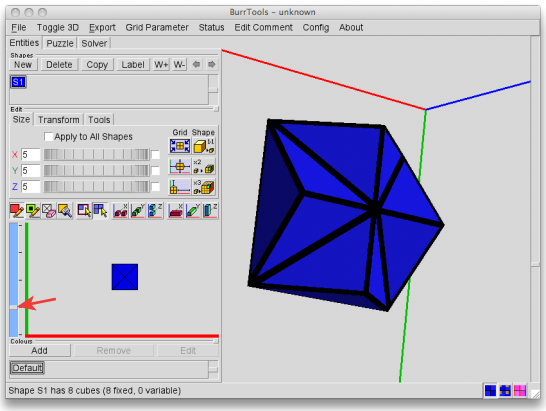




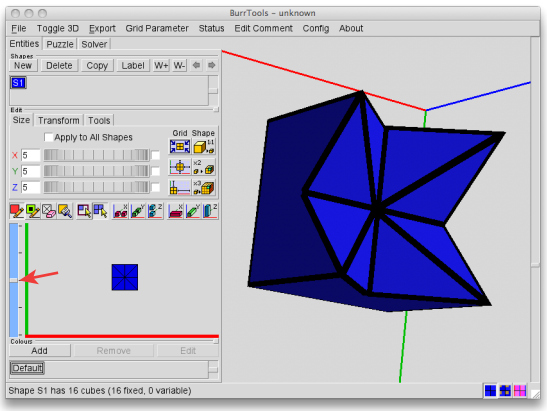
Step 2: Click once in each of the four empty regions of the grid editor -- each will fill with a fixed voxel.



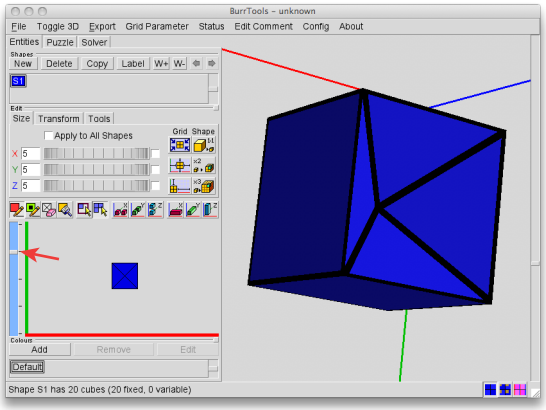
Step 3: Adjust the zoom level of the 3D viewer so that the shape comfortably fills the 3D viewer window. Give yourself some spatial perspective by dragging the mouse in the 3D vewer window to rotate the shape.



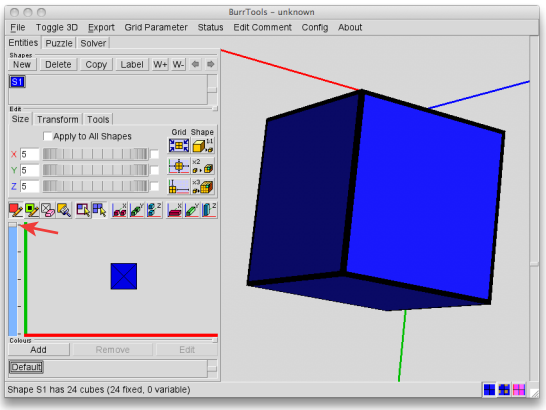
Step 4: Move the grid editor forward one layer, and then click once in each of the four empty regions. Notice how the shape changes in the 3D viewer. Frequently rotating the shape in the 3D viewer will help you maintain spatial perspective of what is happening as you add voxels to the model.



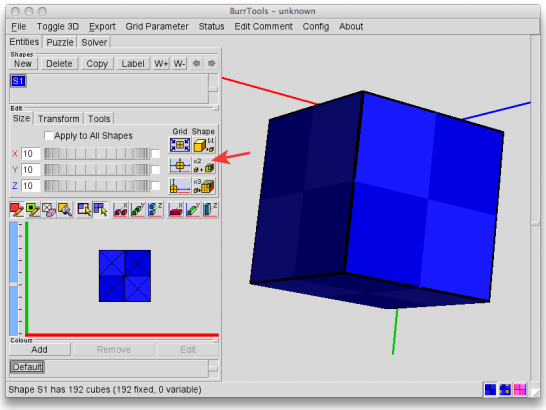
Step 5: Move the grid editor forward another layer. Notice that there are eight empty regions in this layer. Click once in each of these eight regions, paying attention to how it affects the model in the 3D viewer.



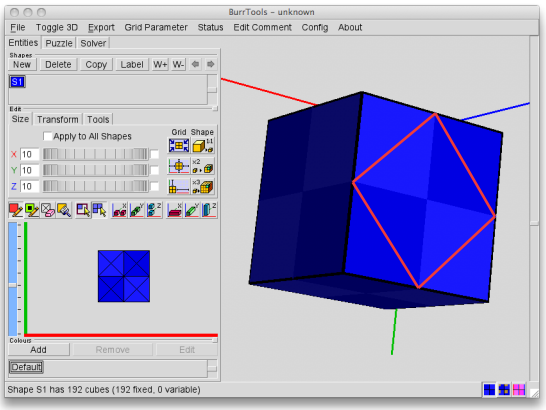
Step 6: Move the grid editor forward another layer. Notice that there are once again four empty regions in this layer. Click once in each of the four empty regions.



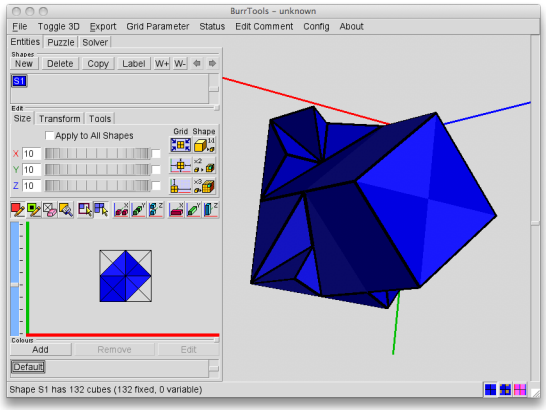
Step 7: Move the grid editor forward one last time to the front layer of the shape. Fill in all four regions to complete the solid 5x5x5 voxel cube.



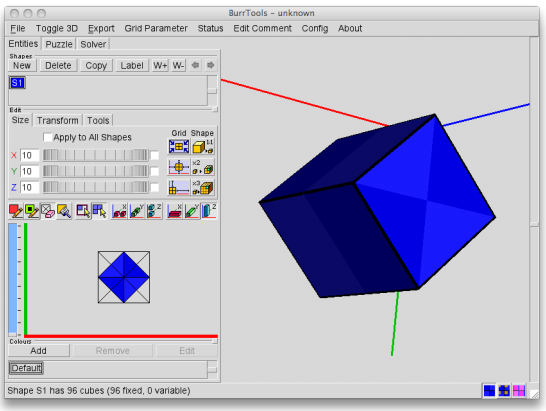
Step 8: Click on the “Double Size” button to turn the solid 5x5x5 cube into a solid 10x10x10 cube. You will need to adjust the zoom level of the 3D viewer again.



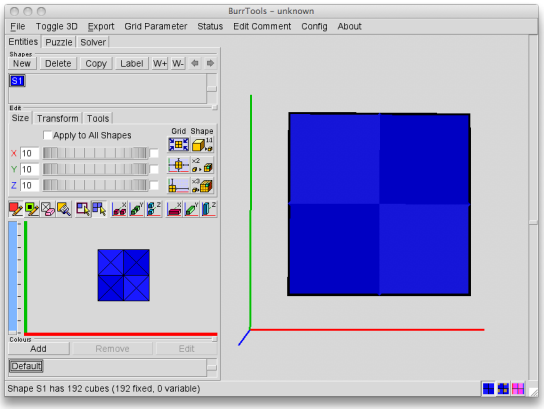
Planning Ahead: Hidden inside of the 10x10x10 voxel cube is a “size one” rhombic dodecahedron. “Size one” means that it is the smallest RD that can be modeled using the BurrTools rhombic tetrahedral space grid, and hereafter will be referred to as a *“1xRD unit”*



The process of removing excess voxels from the 10x10x10 cube is like peeling the skin away from fruit. Notice how the midpoints of each edge are connected, and then used as cut lines for the “peeling” process.

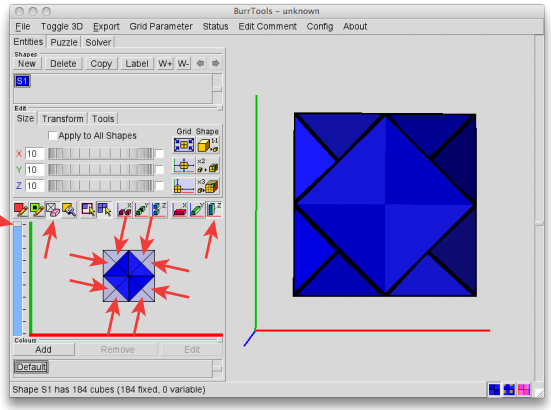


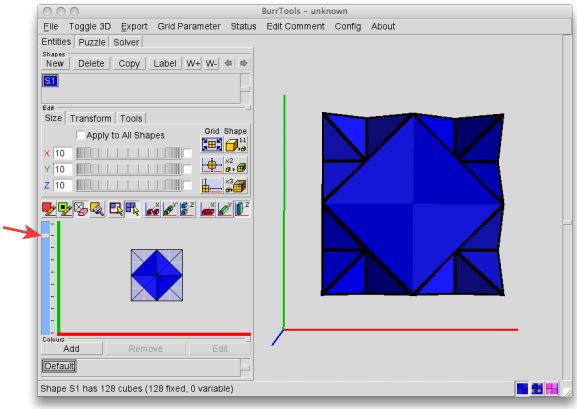
Note: You may also do the peeling in the 3D viewer window by holding the ***CTRL*** button and clicking on voxels that you want to remove. Unintentionally removed voxels may be filled back in place by holding the ***SHIFT*** button instead of ***CTRL***.



Step 9: You have probably got your 3D viewer all jumbled up by now, so take this opportunity to line things up so that you are looking at the front of the cube. Notice how the red and green axes in the bottom left corner of the 3D viewer are aligned with the ones in the bottom left corner of the grid editor.

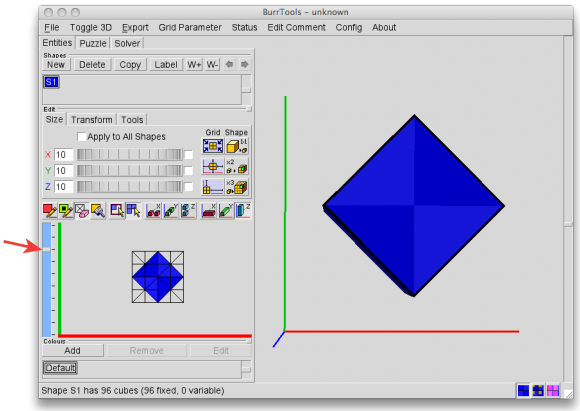
Step 10: Enable the "remove voxels" tool, and also enable Z-axis "layer spanning" mode. Make sure that the grid editor is set to the front most layer, and then click once in each of the eight corner regions, as shown.

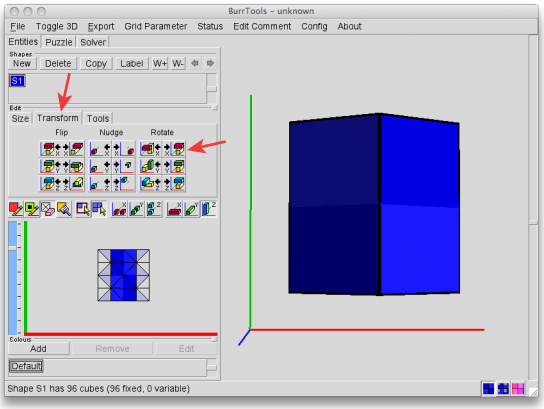




Step 11: Move the grid editor back one layer and then repeat the eight mouse clicks to remove further material.

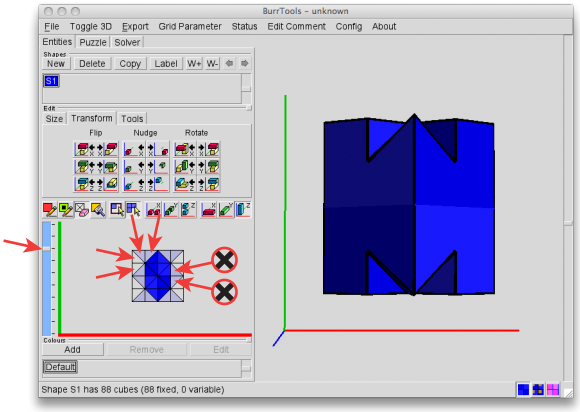
Step 12: Move the grid editor back one more layer. Notice that (as was the case in Step 5) there are twice as many regions to click on this time. *(this happens every five layers...)* Click on these sixteen regions to complete the peeling process for the Z axis of the model.

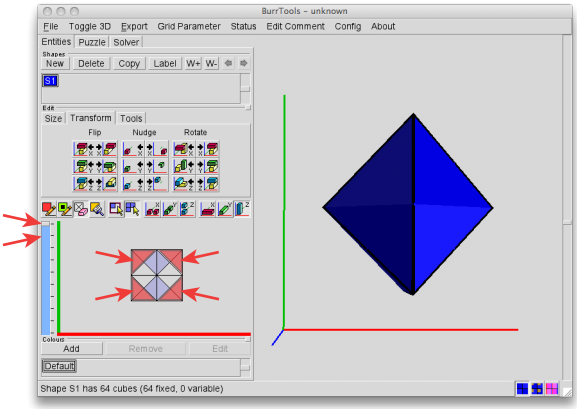




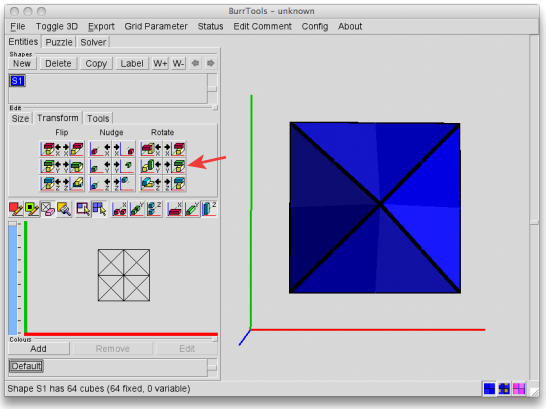
Step 13: The "peeling" process must be done along all three cartesian axes three cartesian axes in order to reveal the 1xRD unit. Rather than changing the tool setup or 3D viewer orientation, instead do a single X-axis rotate transformation.

Step 14: With the model rotated and ready for more voxel removal, go ahead and click on the same 16 regions that you removed in Step 12. Be sure to click on all 16, and not any others.

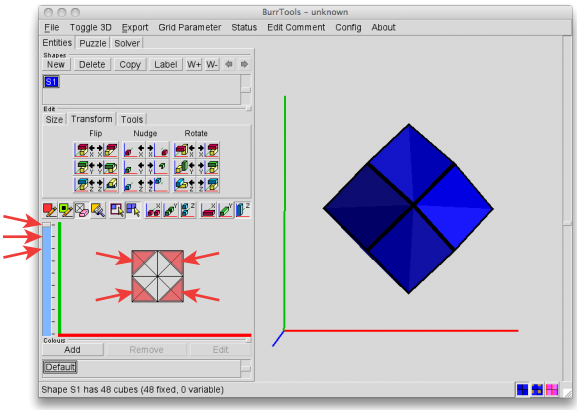




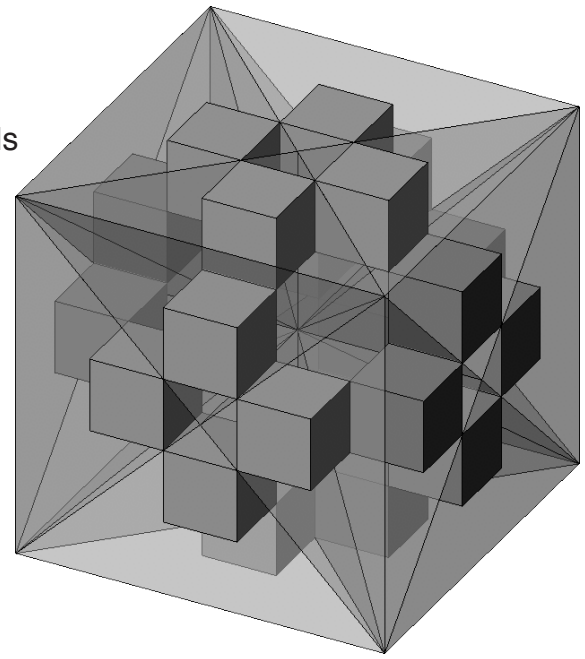
Step 15: Move the grid editor forward one layer and click in the appropriate corner regions to continue along with the peeling process. Doing it again for the front most layer of the model should complete the second round of cuts.



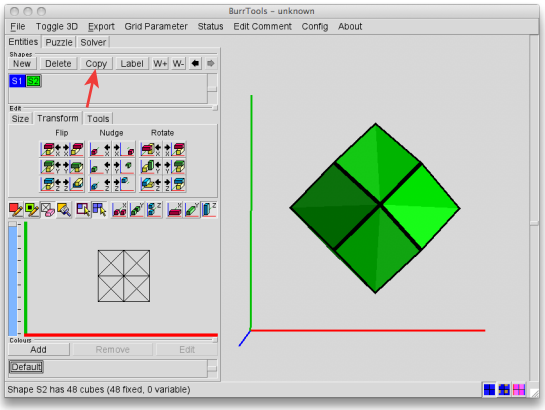
Step 16: All that is needed to set up the model for the third set of cuts is a single Y-axis rotate transformation.



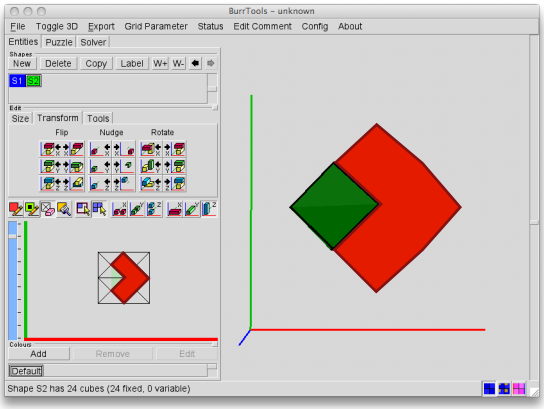
Step 17: Repeat the same sequence of actions on the front three layers of the model to complete the peeling process. Congratulations, you have just created a rhombic dodecahedron!!



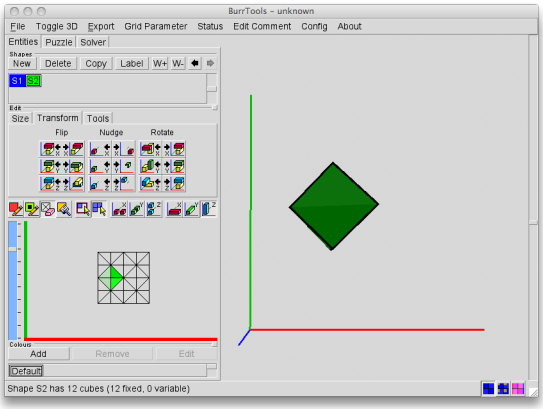
Analysis: If you are already familiar with using BurrTools in *Z-axis layer spanning mode*, then you may be wondering why it was necessary to click on the same (or at least similar) regions on three different layers in order to get the job done. To understand the reason for this, consider the manner in which BurrTools emulates the rhombic tetrahedral space grid onto a cubic matrix of program memory. Notice that most memory cells in the 5x5x5 voxel grid are not used, and also notice that of the 24 which are used, most share their *(rank / row / column)* with at most one other cell in the 5x5x5 grid.



Step 18: Before we go wild cutting up our 1xRD unit into smaller (and more useful) pieces, let's make a copy of it first. BurrTools does not have any sort of ***UNDO*** feature, so it is good practice to frequently make copies of your models whenever you are happy with a series of changes.

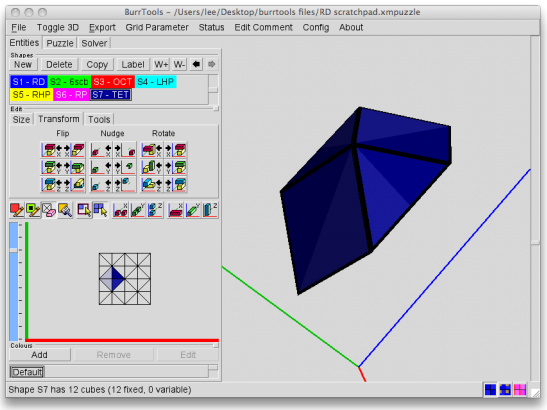


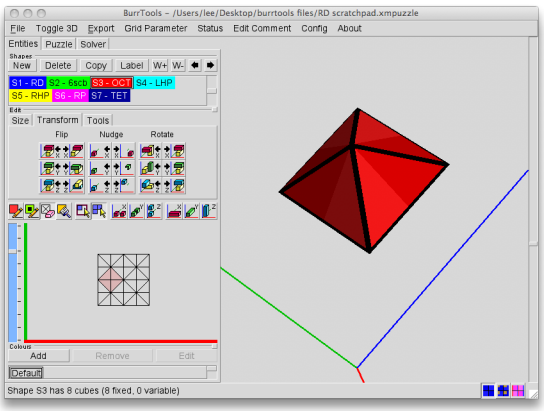
Step 19: Using either the *three-step Z-axis layer spanning technique* in the grid editor, or using *rapid-fire* ***CTRL****-clicking* in the 3D viewer, remove 3/4 of the copy of your 1xRD unit, as shown here.



The remaining quarter of the 1xRD unit is a very useful shape called a *six sided center block*, or 1x6SCB. Rotate the 3D viewer around to get a sense for what this shape looks like, and also switch back and forth between the 1xRD unit and the 1x6SCB to get a sense for how a 1xRD unit can be thought of as four 1x6SCB units put together.

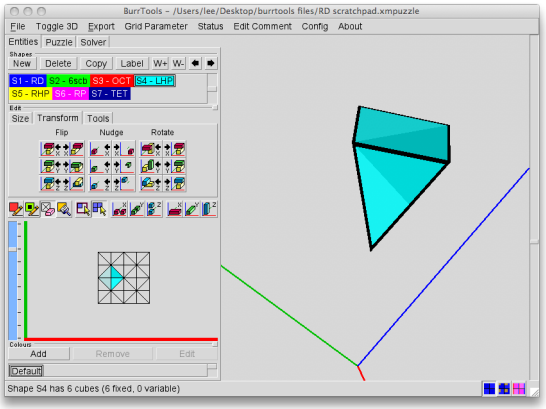
Step 20: If you haven't been using the 3D viewer method of editing the model, then make another copy of the 1xRD unit and practice cutting it into 1x6SCB units by **CTRL**-clicking away the unwanted voxels. Also, try this with various rotations of the model in the 3D viewer -- not just the viewing angle shown in Step 19. Make several more copies of these 1x6SCB units, and try cutting them up into the following shapes:

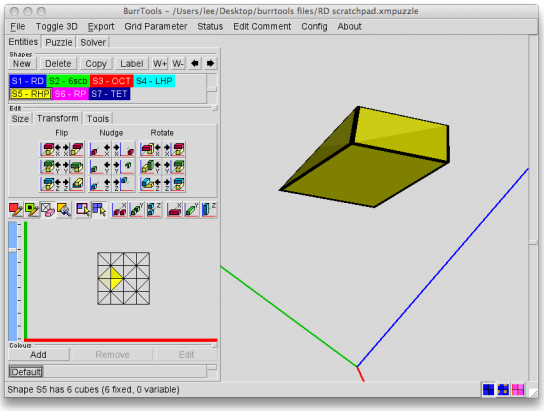




The **Octahedral Block** (or *1xOCT*) is made from a 1x6SCB by removing the pointy ends...

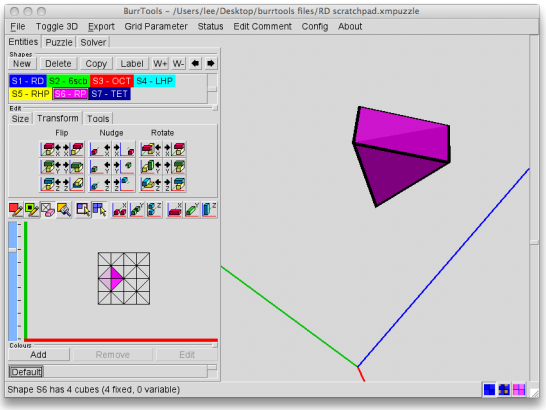
The **Left Handed Prism Block** (or *1xLHP*) is made from a 1x6SCB by cutting it in half along one leg of its central “X” shaped ridge line...

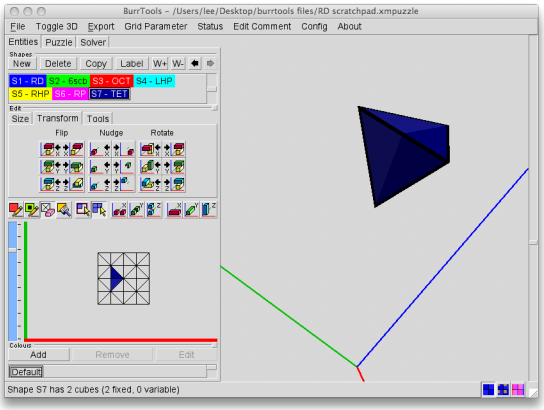




The **Right Handed Prism Block** (or *1xRHP*) is made from a 1x6SCB by cutting it in half along the other leg of its central “X” shaped ridge line...

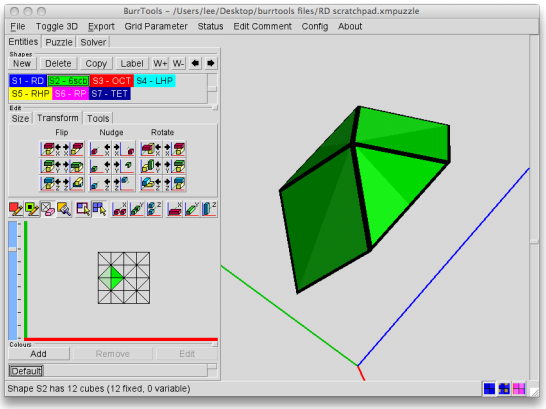
The **Rhombic Pyramid** (or *1xRP*) is made from a 1x6SCB by cutting along both legs of its central “X” shaped ridge line...

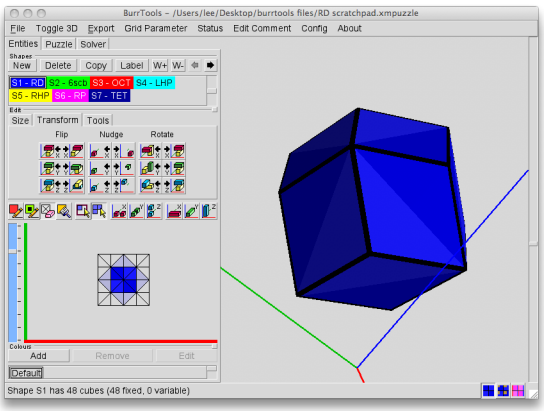




The **Tetrahedral Block** (or *1xTET*) is half of a 1xRP. It is the shape of the pointy tips that get removed from a 1x6SCB to create a 1xOCT. All of the basic shapes listed here can be divided cleanly into tetrahedral blocks like this one...

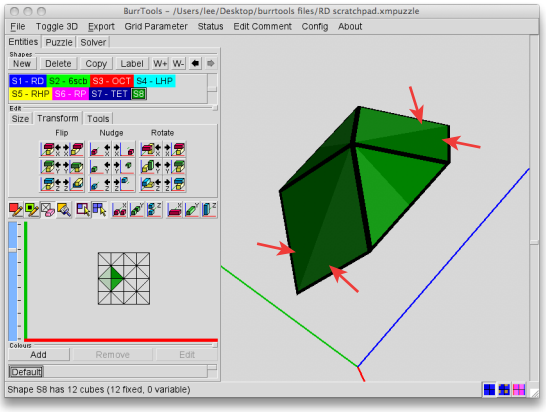
...and as long as we’re listing them all off, here again is the **Six Sided Center Block** (or *1x6SCB*)...

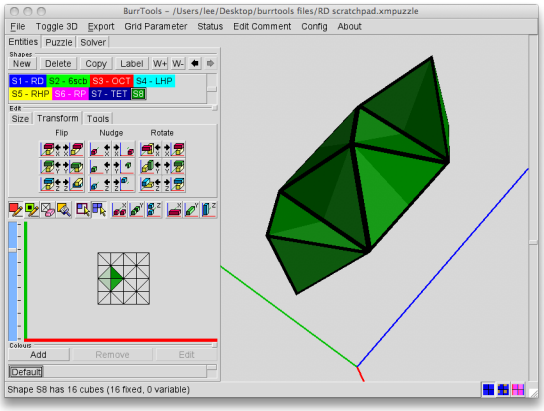




...and the **Rhombic Dodecahedron** (or *1xRD*)

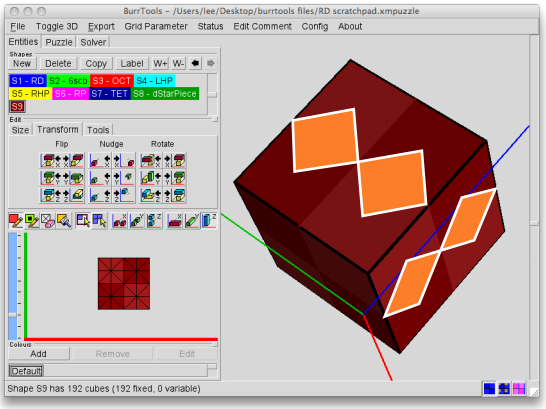
Step 21: Make a copy of a 1x6SCB and then, using the 3D viewer, **SHIFT**-click on the four locations shown here...

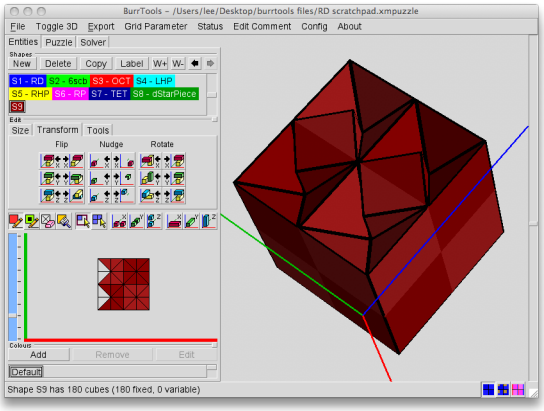




...the resulting shape is one of six identical pieces that make up the ***Diagonal Star*** puzzle.

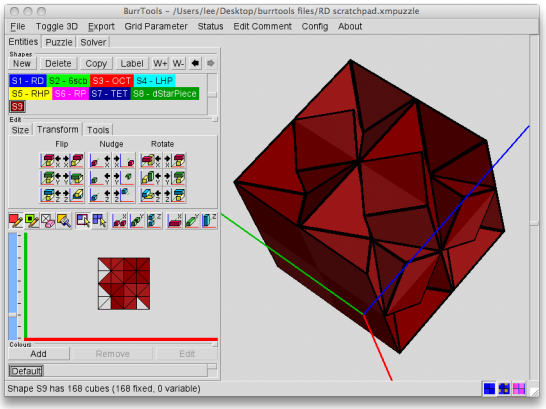
Step 22: Make a solid 10x10x10 voxel cube, and then, using the 3D viewer, **CTRL**-click away the outer layer of two adjacent faces, leaving alone the orange-shaded regions as shown here...

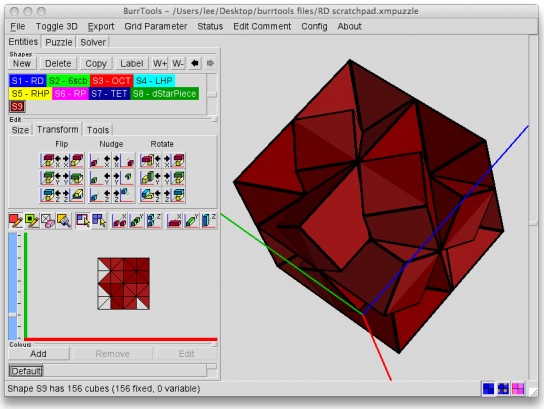




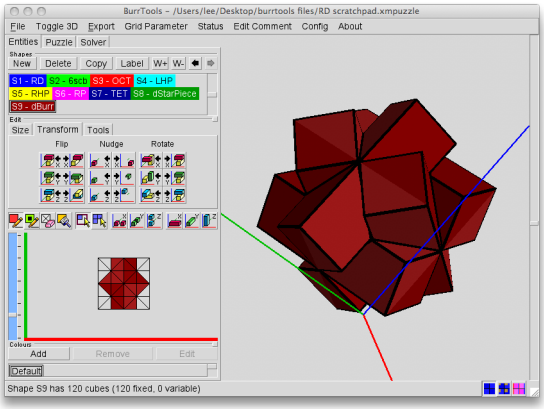
This is what the model looks like after clicking away one of the six faces of the 10x10x10 cube. Exactly 12 voxels have been removed.

This is what the model looks like after clicking away the appropriate voxels on the adjacent face of the 10x10x10 cube. 12 more voxels (24 total) have been removed from the solid cube.

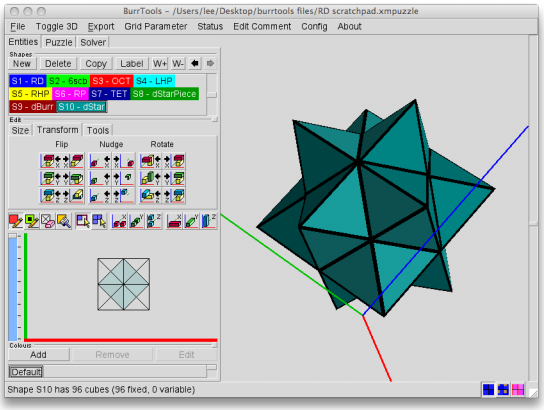




Step 23: Continue removing 12 voxels from each face of the 10x10x10 cube, being sure to do so in a manner that maintains the symmetry of the resulting shape. This is what the model looks like after removing the appropriate voxels from three faces of the cube...

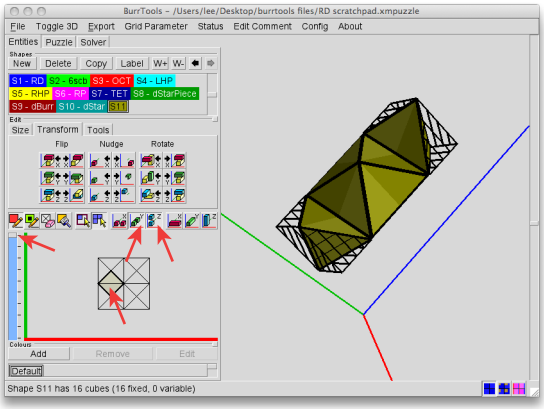


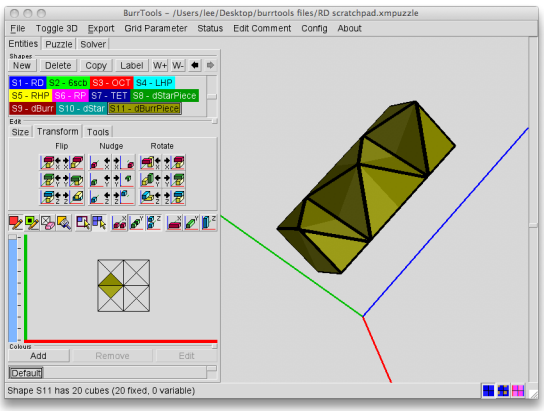
...and this is what the model looks like after all six faces have had the appropriate 12 voxels removed. This model is the assembled shape of the ***Diagonal Burr*** puzzle. "But wait!" you may be thinking to yourself, "Wasn't the puzzle piece that we made for the *Diagonal Star* puzzle, not the *Diagonal Burr*?"



Step 24: Yes, that's true... so let's do both puzzles then. Make a copy of the *Diagonal Burr* that you just made, and **CTRL**-click away the four voxels on each face that you were being careful to avoid during Steps 22 and 23. That's 24 voxels total that you need to remove. The resulting shape is what the assembled ***Diagonal Star*** puzzle looks like... but this shape also has a more technical name: ***The First Stellation of the Rhombic Dodecahedron!!***

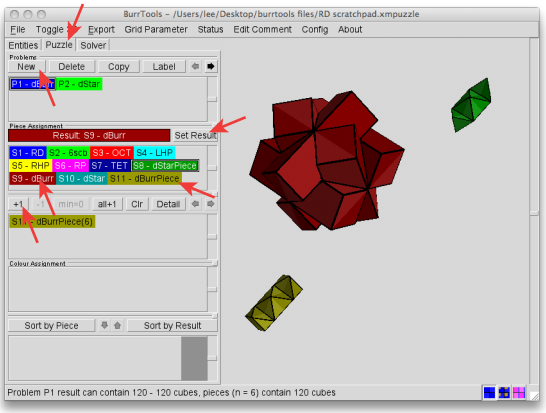
Step 25: Make a copy of the *Diagonal Star puzzle piece*, and then either **SHIFT**-click on the appropriate four voxels as shown here, or you can try this trick with the grid editor which adds all four voxels at once: Make sure that all three layer-spanning modes are disabled, and then enable the Y-axis and Z-axis mirroring modes. Move to either the front or back layer, and then click once as shown. Notice how the 3D viewer displays a wireframe preview of what voxels will be added.

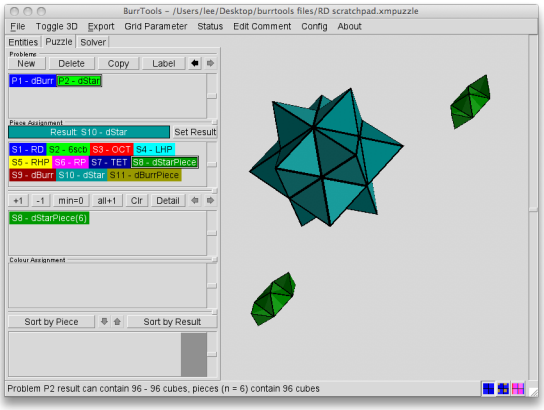




Regardless of what method you choose to add the voxels, the resulting shape is one of six identical pieces that make up the ***Diagonal Burr*** puzzle.

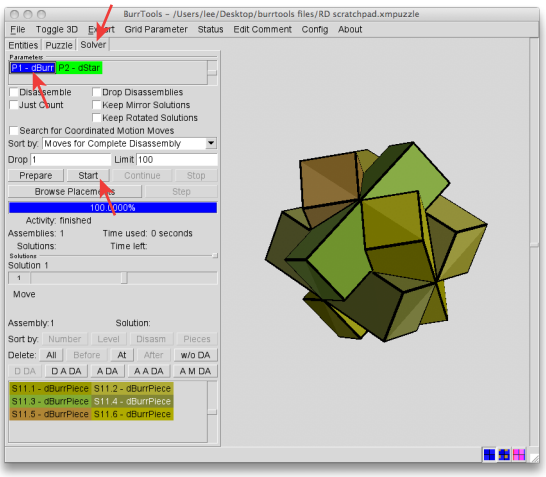
Step 26: Switch over to the **Puzzle** tab and create a **New** puzzle. Select the assembled *Diagonal Burr* puzzle shape and click the **Set Result** button. Then select the *Diagonal Burr puzzle* piece shape and click the **+1** button six times.

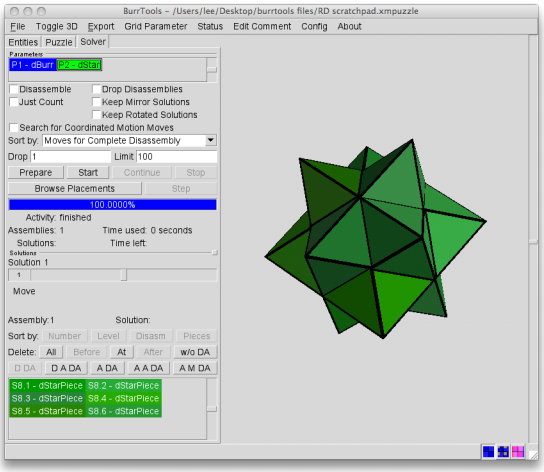




Repeat this process for the *Diagonal Star* puzzle, starting out by defining another **New** puzzle, then setting the result shape, and lastly adding six copies of the *Diagonal Star* puzzle piece shape.

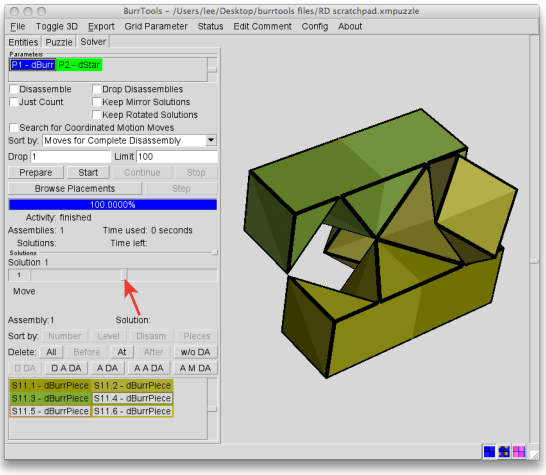
Step 27: Switch over to the **Solver** tab and select the *Diagonal Burr puzzle* **Parameters** (P1) that you defined in Step 26. Click the **Start** button to get things going.

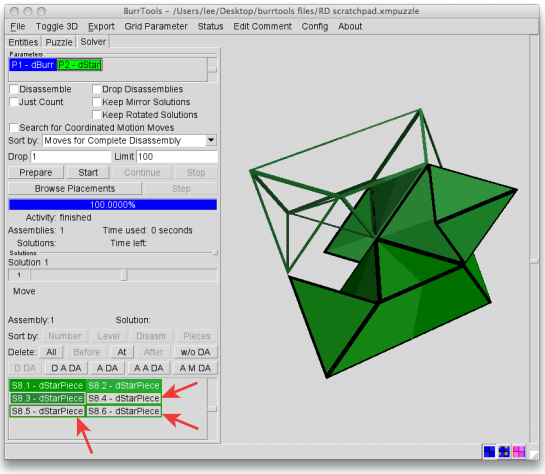




Repeat these same steps (Select **Parameters**, then click **Start**) to analyze the *Diagonal Star puzzle* (P2).

If solutions were found, you can browse through them by dragging the **Solutions** slider.

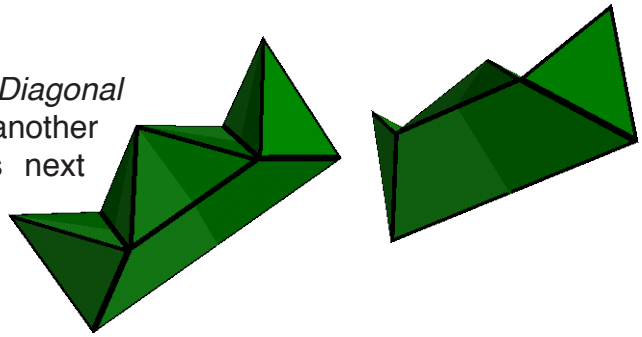


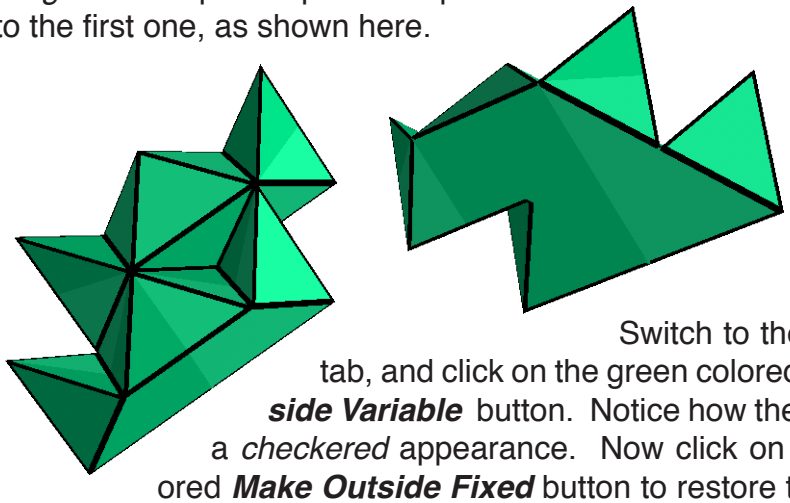


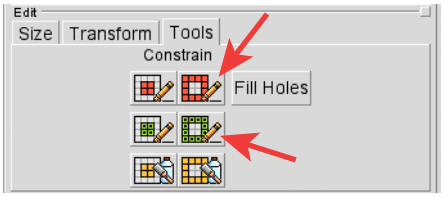
Rotate the model in the 3D viewer, and toggle the visibility of individual pieces to help visualize and understand the solution.

Advanced Analysis:

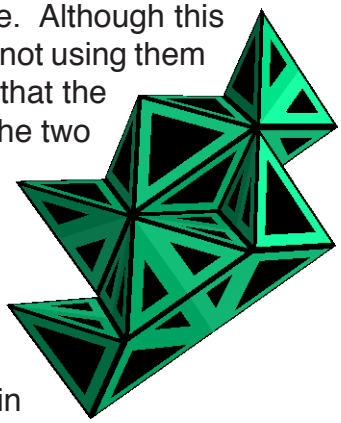
Switch back over to the **Entities** tab and create a copy of the *Diagonal Star* puzzle piece. Now, using the 3D viewer, **SHIFT**-click another *Diagonal Star* puzzle piece shaped cluster of voxels next to the first one, as shown here.

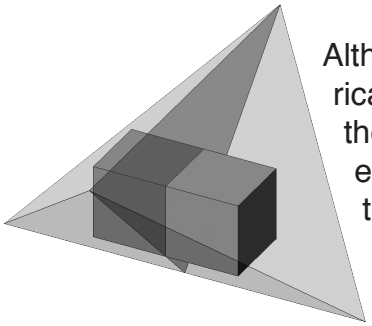




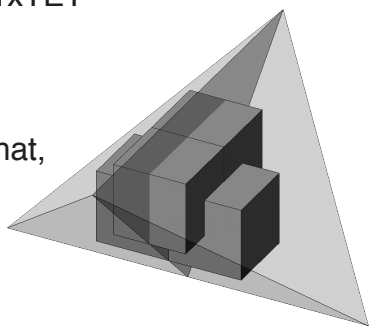


Switch to the Tools sub-tab, and click on the green colored ***Make Outside Variable*** button. Notice how the model gets a *checkered* appearance. Now click on the red colored ***Make Outside Fixed*** button to restore the original appearance. Although this technique of looking at the way that BurrTools renders variable voxels is not using them for their intended purpose, it is nonetheless a useful tool for analyzing the way that the voxels are clustering together. Notice that when seen with a variable exterior, the two copies of the Diagonal Star puzzle piece shown here obviously have different clustering patterns. Could this cause problems?

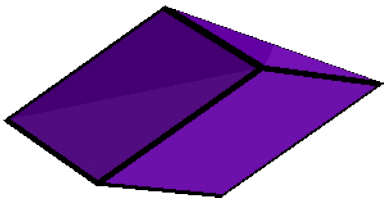




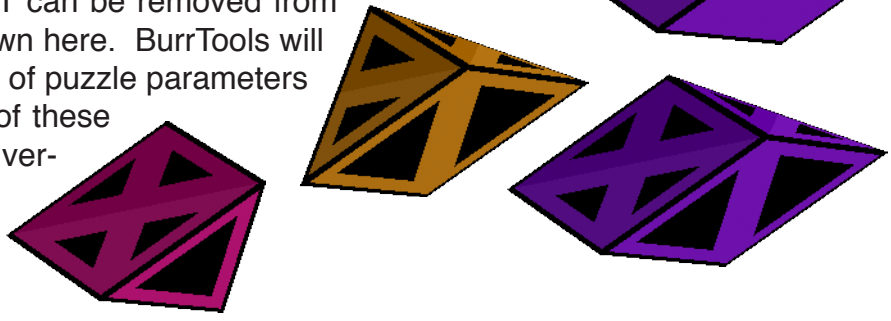
Although these two different tiling patterns are geometricaly equivilant when considering the solid shape that they represent, they are **NOT** equivalent when considering the way that the cluster of voxels are arranged in the cubic matrix of program memory. Consider the 1xTET memory emulation diagram on the left.

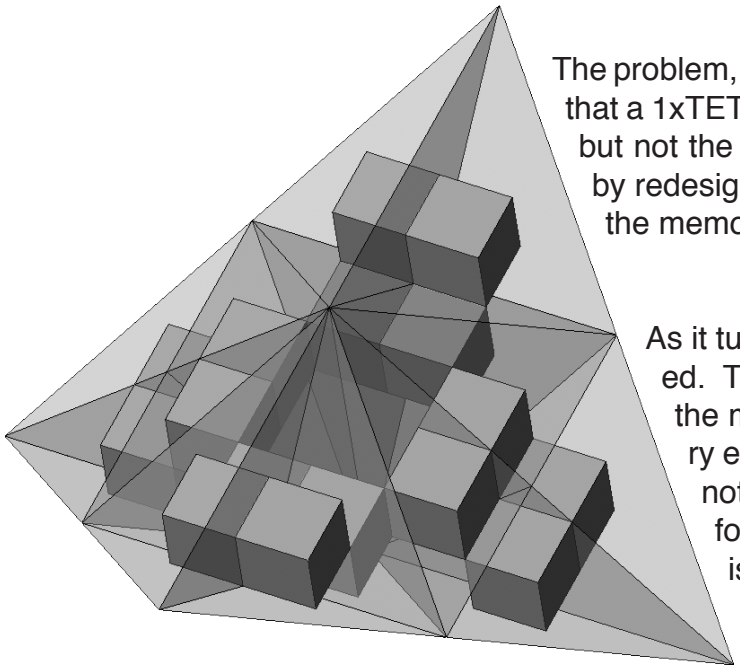


The shape of the 1xTET has symmetry such that, when the shape is flipped over and turned sideways, it occupies the same cross section of space as before. The **BIG PROBLEM** here is that the BurrTools *right-angle-tetrahedron* memory emulation method is **NOT** so friendly about flipping and rotating, as seen in the diagram on the right.



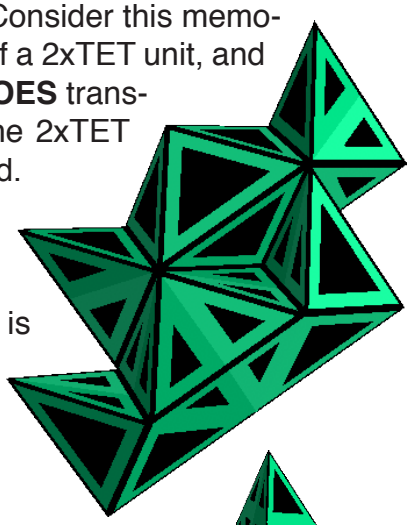
It turns out that yes, it really is quite easy to encounter this problem as you are working on a rhombic tetrahedral space grid project with BurrTools. Consider the two ways that a 1xTET can be removed from a 1xLHP to produce a 1xRP, as shown here. BurrTools will not find a solution if you create a set of puzzle parameters (see Step 26) attempting to fit one of these versions of the 1xRP into the other version as the result shape.



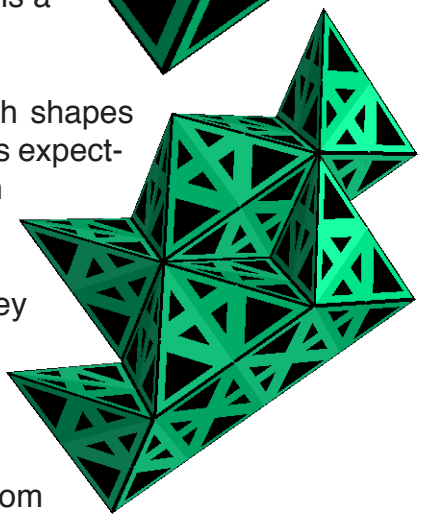


The problem, it seems, is that the shape of a single voxel is such that a 1xTET may be divided in half along one of its 90º edges, but not the other. Perhaps this complication may be solved by redesigning the basic voxel shape, or perhaps modifying the memory emulation of the existing voxel shape?

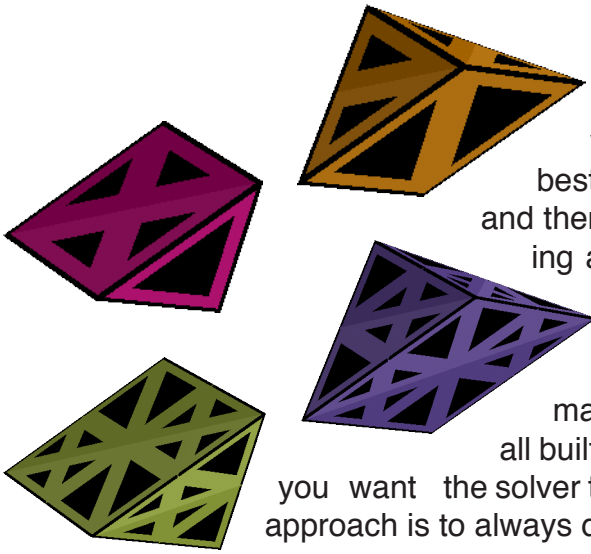
As it turns out, such drastic measures may not be needed. The problem resolves itself nicely when the size of the model is doubled. Consider this memory emulation diagram of a 2xTET unit, and notice that this grid **DOES** transform cleanly when the 2xTET is flipped and rotated.



Further contemplation of the situation will lead you to realize that this is not the only way to emulate a 2xTET... if each of the eight 1xTET units contained within the 2xTET were individually flipped and rotated, the resulting emulation would look quite diferent from this one, yet it too is a valid way to tesselate 2x RD geometry.



Is this a problem? Yes, insofar that you still cannot mix-and-match shapes made by the two 2x tesselation methods in a set of puzzle parameters expecting to get good results... On the other hand, if all of the shapes in a set of puzzle parameters are built using the same 2x tesselation method, then the BurrTools solver has no problems, and we get the results that we expect. In an effort to eliminate problems before they ever come up, we have designed the **Double Size of Shape** button to always build 2x shapes using the same tessalation method.



Considering this matter from the standpoint of how it affects your user experience, we decided that for now it would be best to let you build any shapes that you want in 1x geometry, and then leave it up to you to do the 2x transformation before defining any puzzle parameters and running the solver. The benefit of doing it this way is twofold: firstly, editing shapes in 1x geometry involves eight times less clicking that editing 2x geometry... and secondly, it is possible to run analysis on many (but not all) puzzles using 1x geometry if the shapes are all built using compatible tesselation methods. This can be useful if you want the solver to run as fast as possible... but for most applications, the best approach is to always double the size of your shapes before running the solver.