

# SceneGen User's Guide

SceneGen is a program that enables rapid development of complex terrain-based environments for engineering simulations. It integrates CAD-generated solid models with sampled soil data and other geometrically defined objects into a single environment. From this environment, different geometries can be exported for mesh generation, line of sight calculation, visualization, etc.

SceneGen has been designed to be multiplatform. It was written with ANSI compliant C++ and uses the Qt interface library. It has been compiled and runs on Windows XP, SUSE Linux, and Mac OS X.

While SceneGen deals with CAD objects, it is not a full CAD system itself. Instead, it is a simple object orientation and placement tool. This is to simplify the interface for users who just want to design a simple scene from existing designed or sampled objects. Currently, it exports data for the Omicron program, developed at ERDC-ITL, and for ERDC-EL's vegetation program.

## Supported Formats

SceneGen works with its own file format, the OSDL (Object-based Scene Description Language). This file format is a text format that stores the names and orientations of CAD models, heightfield data, and other geometries (plants, buildings, etc.) It can import data from some FEM models (Omicron and Veg models, at present). It can export code for POV-Ray and Wavefront OBJ files for visualization.

### Supported CAD Models for Input

- 2DM (BYU 2D data)
- 3DM (BYU 3D data)
- STH (HYSSOP directed poly format)
- Poly (ADH standard polygon format)
- PTS (points file)
- TXT (points file)
- OBJ (Wavefront OBJ file)

## The Scene Generation Process

Import Surface -> Define Areas of Interest -> Import Objects -> Arrange Objects

The typical definition of a scene to be meshed has an area of interest for meshing based on a surface location. The surface itself separates the soil from all other materials,

gaseous or liquid. Multiple layers can define different material strata, but the primary surface is the soil boundary. In most cases, objects exist on or below that boundary.

We use this natural definition of a scene to define the user methods for creating a scene in SceneGen. The image shows a flowchart of user interaction to create a scene.

## OSDL: The Object-based Scene Description Language

The Object-based Scene Description Language is a file format used for storing a scene description. In version 1.1, the OSDL description looks like this:

```
OSDL
{
    version 1.1

    %Omicron_Input mctest.out.txt
    %Veg_Input smooth_small_mines_veg.out.txt
    %Omicron_Output mctest.out2.txt
    %Veg_Output smooth_small_mines_veg.out2.txt
    %POVRay_Output mctest.pov
    %Wavefront_Output mctest_wavefront.obj
}
```

The OSDL language is a tag based language, and will be migrated to XML in future versions, and stand alone from input formats for other programs. In the file format as it now stands, data are defined by the Omicron file input and the Veg model input of Jerry Ballard.

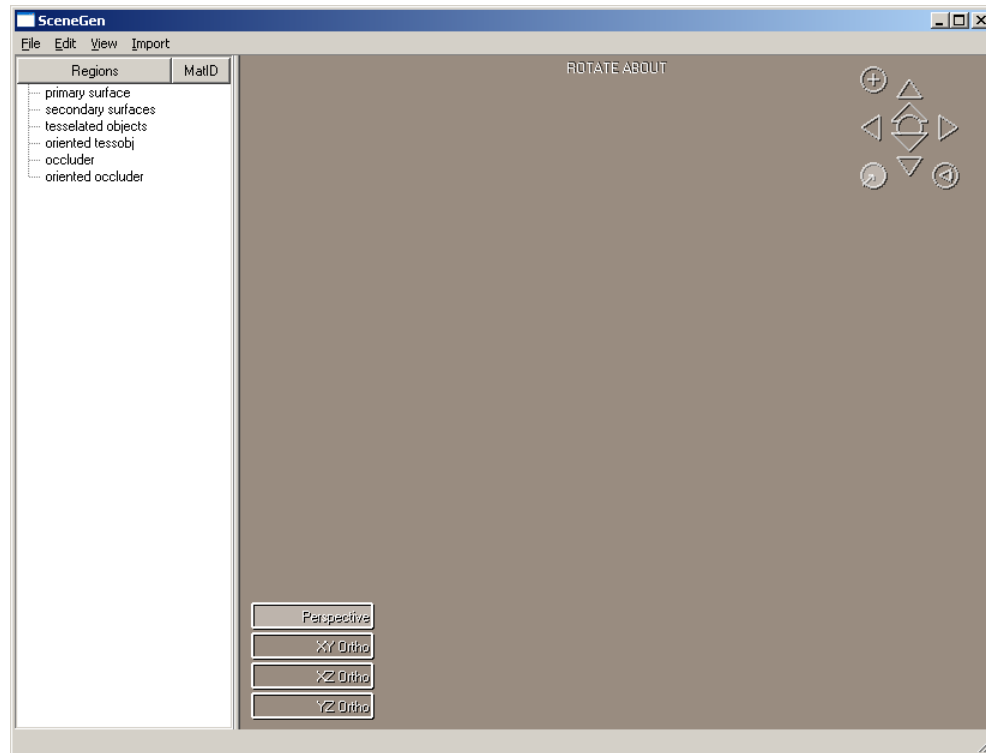
The data outputs are also defined in the OSDL file, and each possible output is defined in the example above.

## OSDL Orientations

A key point to note is that OSDL uses a right handed coordinate system with up being along the positive Z direction. The positive X direction is to the right in the image and the positive Y axis goes into the page.

Objects combined with the scene are expected to use this coordinate system. An additional caveat is that the Objects must be defined with their “front” facing along the positive X axis.

## SceneGen User Interface



**Figure 1. User Interface at startup**

The SceneGen user interface takes data input via an OSDL file and allows objects to be added and/or reoriented in the scene. Figure 1 shows the screen when SceneGen starts. The screen has a menu and the rest is divided into two areas, the Region Area and the Visualization Area.

The Region Area shows the possible types of data currently handled by SceneGen. The primary surface is the surface between soil and another medium. Secondary surfaces define the strata of soils beneath the primary surface, and other media above the surface. Secondary surfaces are not being handled right now by Omicron.

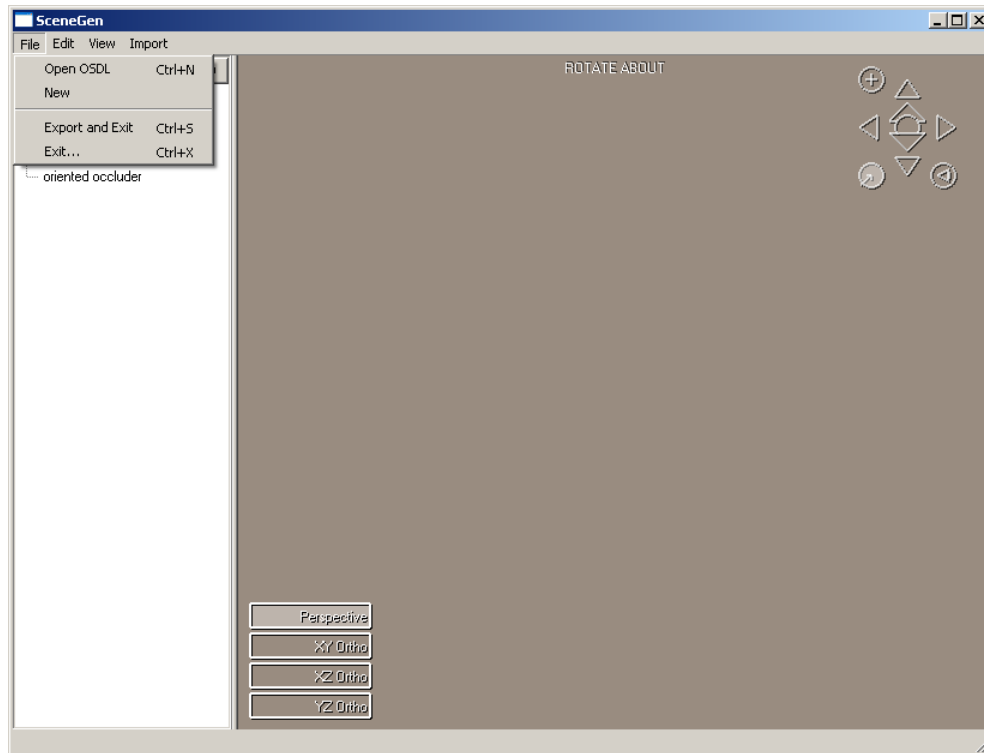
Tessellated Objects are objects that have been created with a CADD program. These objects have a triangulated surface that encloses an interior volume, and is therefore capable of being meshed. These objects can be used by Omicron.

Oriented Tessellated Objects are tessellated objects that are always oriented the same way vertically. This means that the top of the tessellated object always points up. While these can be used in Omicron, they are not explicitly defined by the Omicron file format.

Occluders are triangulated objects which do not have an interior volume, and as such are not capable of being meshed themselves. They are used primarily as methods of determining “visibility” between objects, such as the sun and the ground in a raytracing application. They can also provide 2D face conditionals for 3D elements. At the present, these are not used by either Omicron or Veg.

Oriented Occluders aligned vertically in the same way as Oriented Tessellated Objects. Plant models used in Veg modeling are defined as oriented occluders. This makes sense because plants always grow skyward. Oriented Occluders are used by the Veg model.

## The File Menu

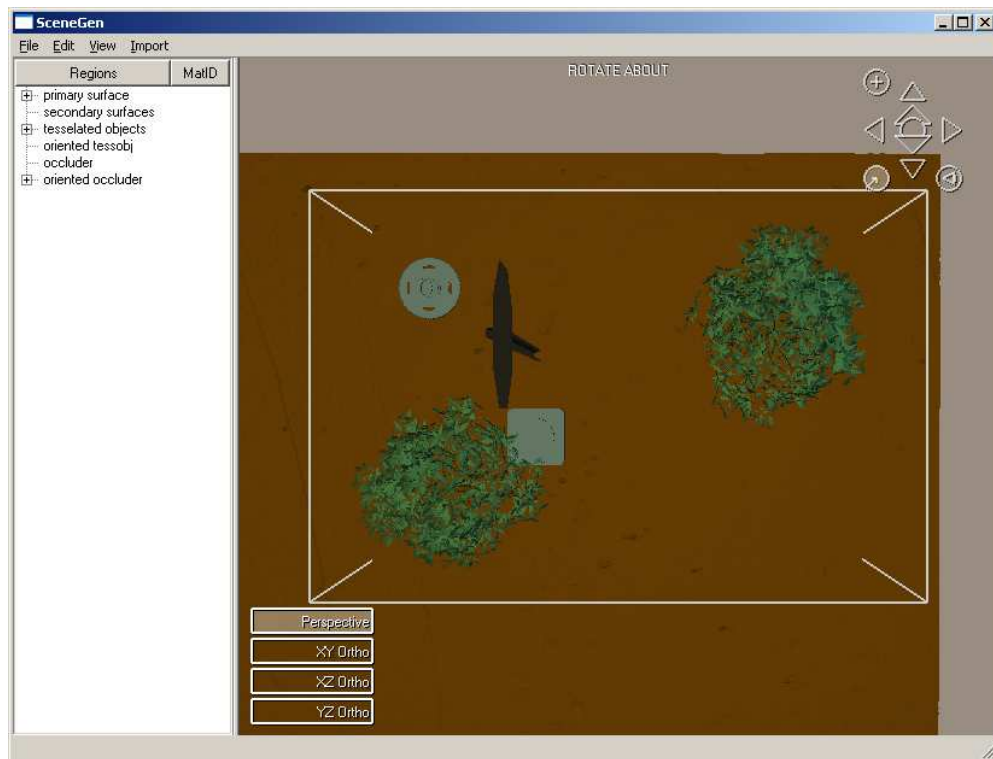


**Figure 2. The File Menu**

The file menu at present looks a lot like a standard File menu. It allows for opening files, flushing for a new project, and ending the program. But for SceneGen, the Open OSDL button allows the user to select an OSDL file, which is defined by Omicron and Veg input files and requires various incidental files such as surface LiDAR data and object definition files. When an OSDL file is opened, these files must be positioned correctly relative to their specification in the other files. It is intended that Future versions of SceneGen will be able to collect these files. Future versions of SceneGen may not be reliant on Omicron and Veg, either. This means that future versions of the OSDL file will be able to be exported.

The Export and Exit option tells SceneGen that the outputs specified in the OSDL file need to be written out. Each format is written with the filename specified in the OSDL file.

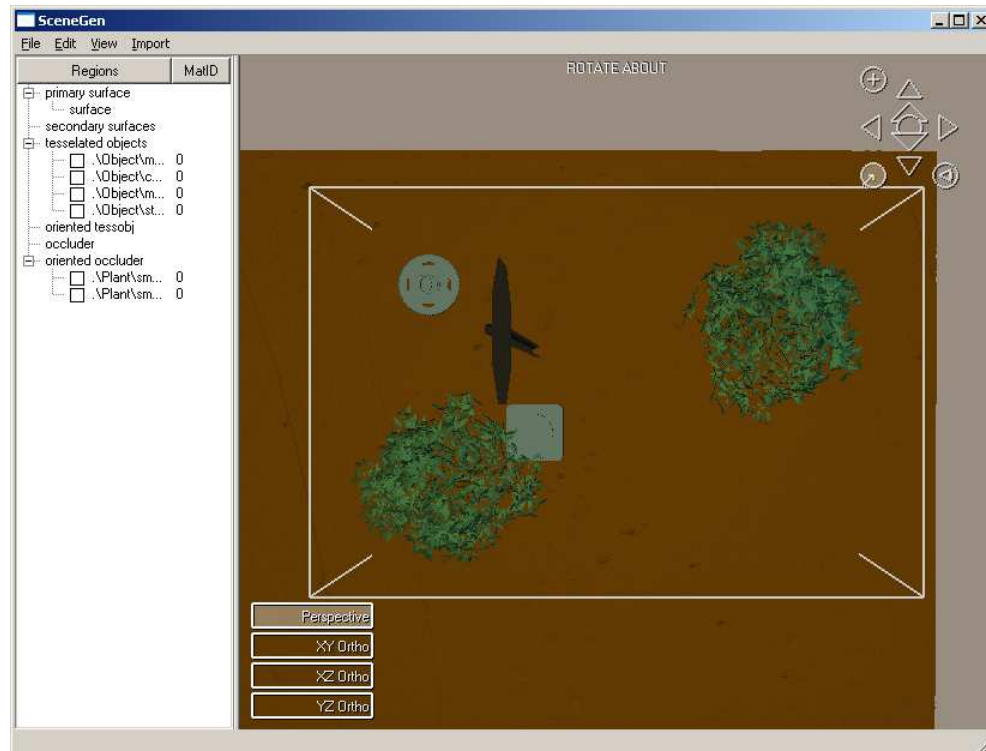
Figure 3 shows a scene that has been loaded from an OSDL file. In this file, there was a primary surface, four tessellated objects (although one would really be classified as an occluder - but it is read from an Omicron file), two oriented occluders (plants), and an area of interest. The region area tree has been updated with the additions.



**Figure 3. After opening an OSDL file**

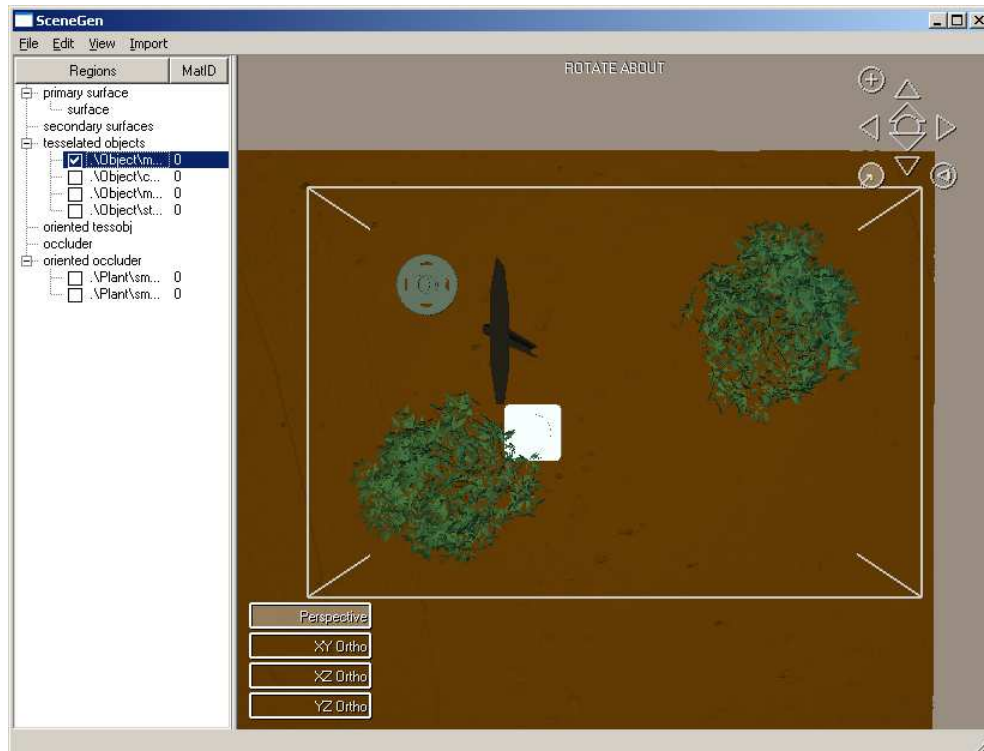
## The Region Area

Figure 4 shows a fully expanded Region tree. The top of the tree shows the type of objects in the file. Each lower level item is an actual object or boundary in the scene. For Omicron and Veg files, the object name is the name of the file containing the object geometry. If an item is selectable for orienting/repositioning, then a checkbox starts its entry in the tree structure. If it is possible to assign a base material ID to an item, then its material ID appears in the MatID column.



**Figure 4. Expanding the Region tree**

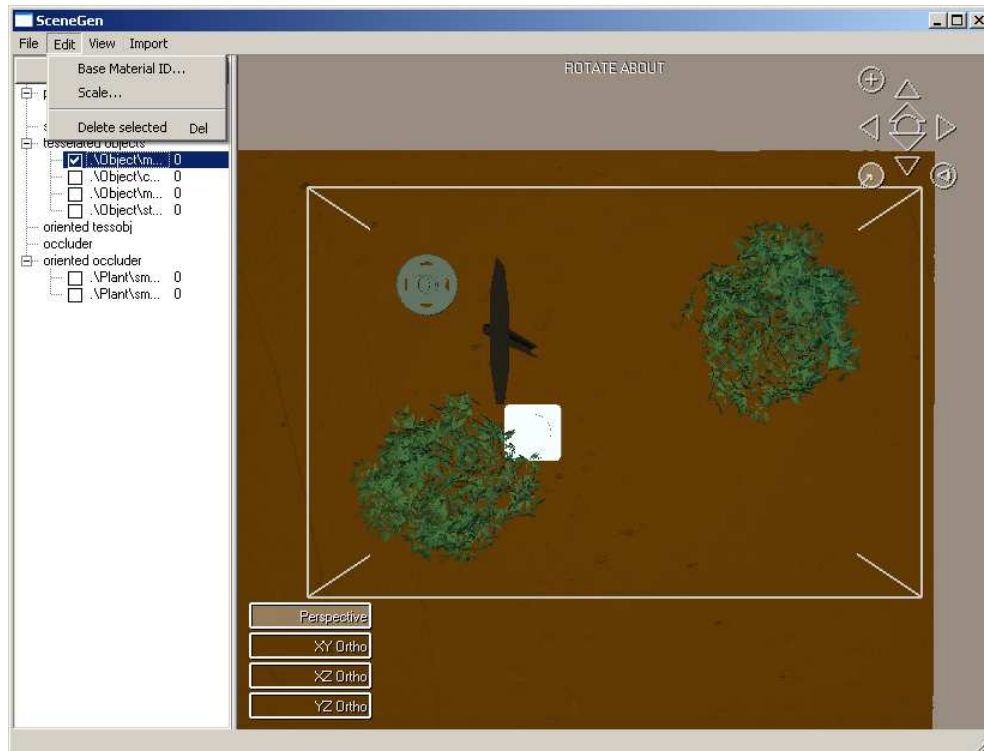
It is possible to select items from the Region Area. Simply check the checkbox next to its entry in the tree, as seen in Figure 5. The item selected is highlighted in the Visualization Area to confirm the selection.



**Figure 5. Selecting from the Region Area**

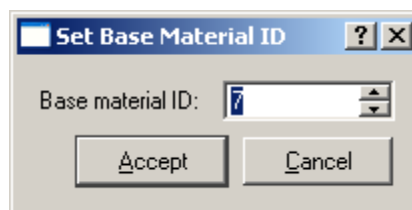
## The Edit Menu

When items are selected, either from the Region Area or the Visualization Area, those items can be edited. The Edit Menu provides a few options for editing objects. From this menu, the user can select the Base Material ID of an object, scale an object, or delete it from a scene entirely.



**Figure 6. The Edit Menu**

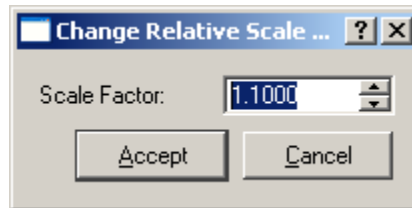
Because future versions of SceneGen will work with object that can possibly have two or more material types in their description, it was decided that rather than define each material ID individually, it would be better to assign a base material ID and each subsequent ID would be consecutively numbered or incremented internally. This would simplify the material assignment process. The dialog in Figure 7 shows how the base material is assigned. Choosing Accept from this dialog assigns the shown material Base ID to all selected objects. At this time, neither Omicron nor Veg use this feature.



**Figure 7. The Base Material Dialog**

In a similar fashion, it is possible to scale any selected object. Choosing “Scale...” from the edit menu displays the dialog box shown in Figure 8. The scale shown in the dialog is the scale of the object prior to this menu item being chosen. In other words, if the scale is set to 1.0 then the items will stay the size they were before the menu item was chosen. Accepting any other value scales the object. This option is available for either Omicron or Veg.



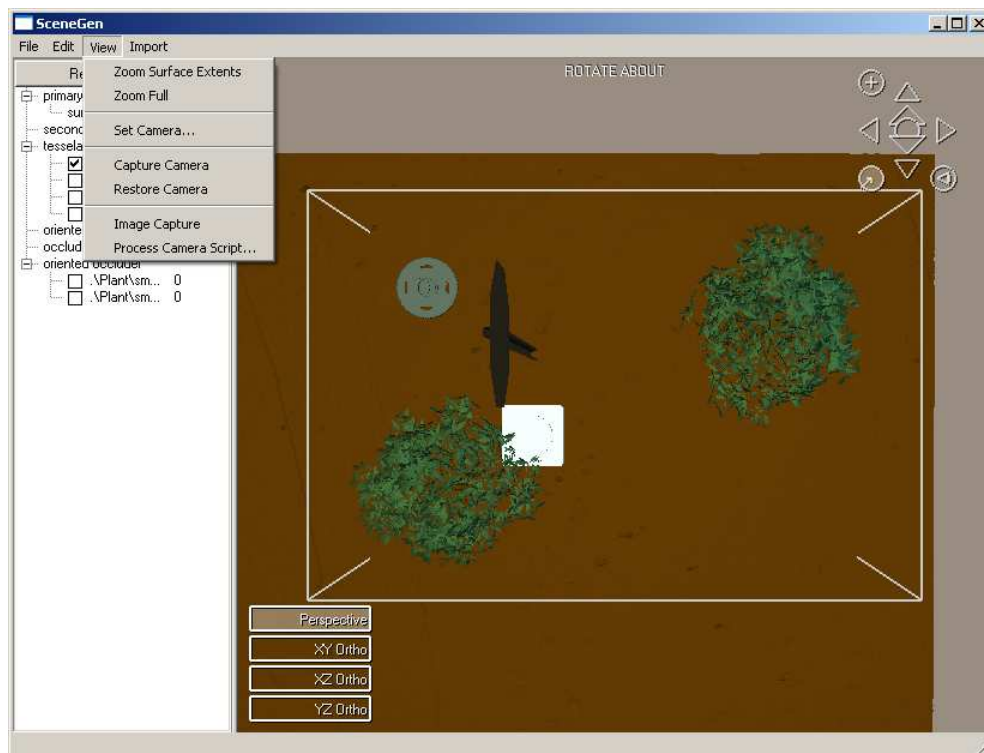


**Figure 8. The Scale Object Dialog**

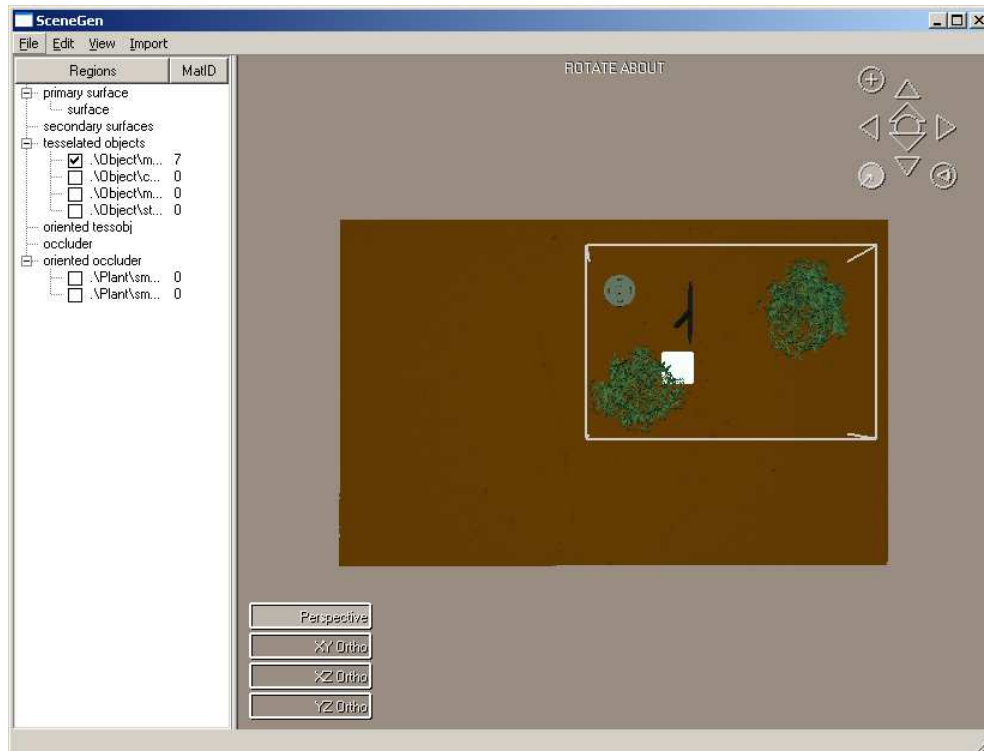
It is also possible to delete objects from a scene in SceneGen. When the menu option is selected or the Delete button pressed, selected objects in the scene are removed. This option works for Omicron but because of conflicts with the Veg format, may cause errors with the Veg file output.

## The View Menu

The View Menu provides several powerful visualization options for the user. The user can choose whether to view the Scene considering the entire extents of the surface data (Figure 10) or primarily focusing on the area of interest (Figure 9).

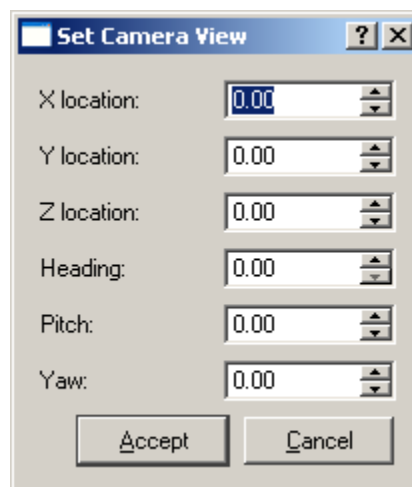


**Figure 9. The View Menu**



**Figure 10. Viewing full surface extents**

If a user knows the specific camera position and orientation that he would like to view the model from (though not very likely), he can enter the coordinates from the “Set Camera View” dialog, from the “Set Camera...” menu option.



**Figure 11. Set Camera Position and Orientation Dialog**

If the user has an interesting camera view that he would like to save for viewing again, he can choose the “Capture Camera” menu option. This option only stores one

view at a time, so its usefulness is rather limited. Choosing “Restore Camera” restores the captured camera position and orientation.

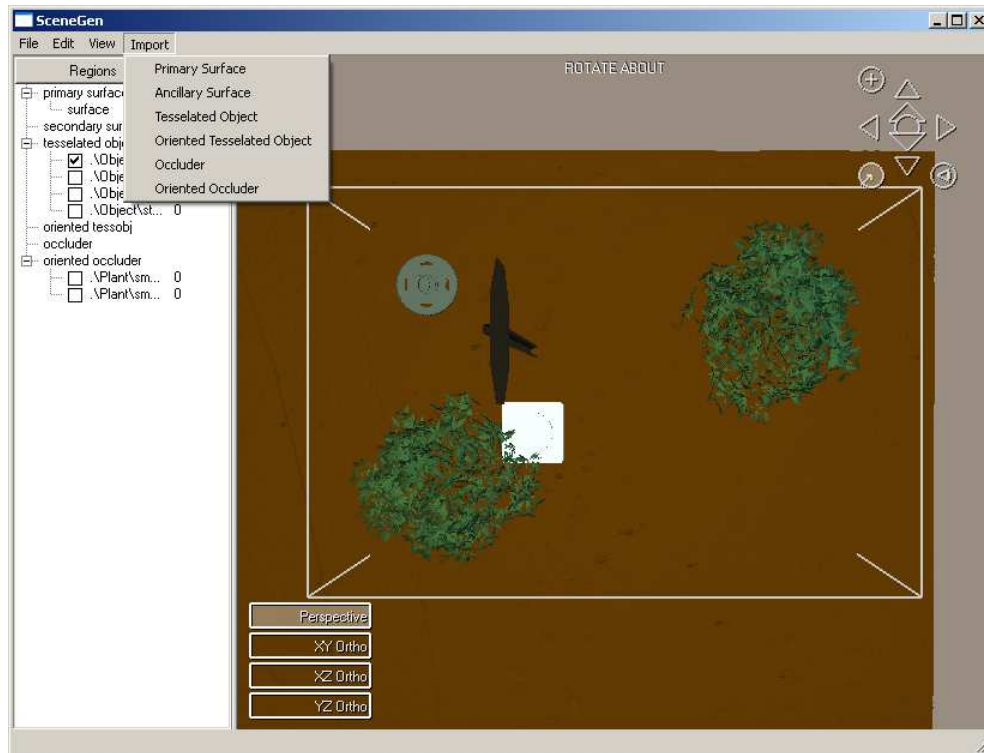
The “Image Capture” menu option performs a bitmap capture of the scene in the Visualization Area. All extraneous overlays are removed. A file dialog appears for selection of the bitmap file name.

The “Process Camera Script” menu option allows the user to capture a sequence of images given a script file. These images are stored in the Microsoft BMP format for lossless compression. The script file is a text file with an extension of “.spt”. The first line in the script file has a string that is the prefix for all the output bitmap filenames. For instance, if the prefix is “scap”, the resultant file names would be “scap0000.bmp”, “scap0001.bmp”, etc. The second line contains an integer which is the number of camera positions and orientations to capture. There must be at least that number of lines following with camera positions and orientations. Each camera position and orientation line has six floating point values – the X location, Y location, Z location, Heading angle, Pitch angle, and Yaw angle for the camera. Each image will be captured in the same fashion as the “Image Capture” menu option – sans overlays.

## **The Import Menu**

The Import Menu is a new addition to the SceneGen Modeler. The options allow the user to add new items to a scene, and provide some information for the input object. These options are included at present, but are not entirely compatible with either Omicron or Veg. Importing surfaces, for instance, does not comply with either program.

Importing surfaces is as easy as selecting a file for import. At present, these surfaces are brought in and offset so that the lowest value corner is at the origin. There is no scale or other deformation applied. Future work will allow us to deform, scale, and translate our surfaces, as well as define the material ID beneath the surface.



**Figure 12. The Import Menu**

Objects can be imported at any time. Imported Tesselated Objects are added to the Omicron file. Because of file problems with Veg, imported objects are not added to the Veg files at this time.

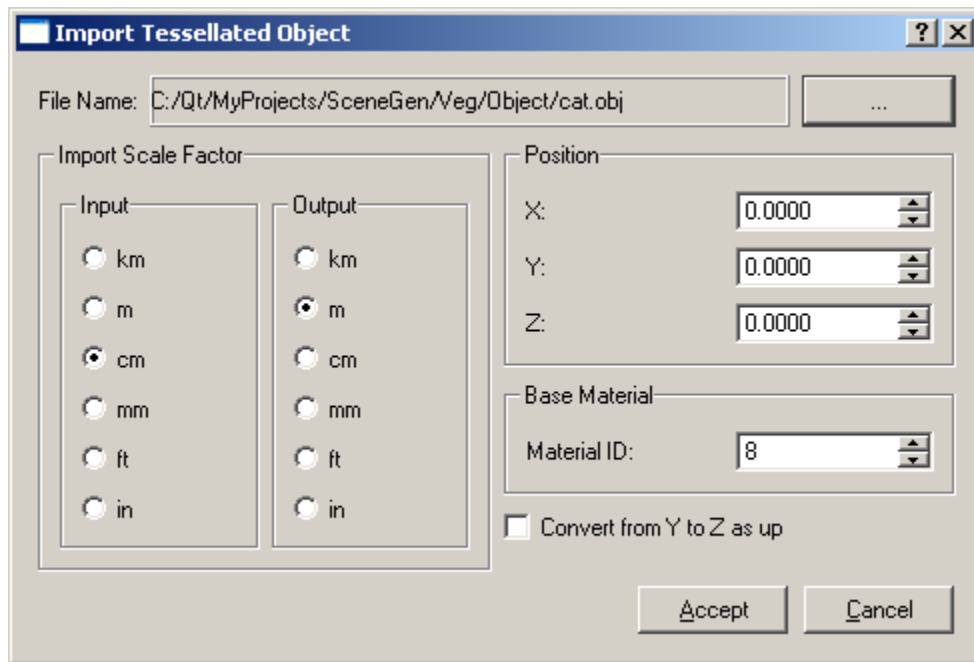
Objects are added to the Scene using the Import Object Dialog in figure x. The same approach is used whether the object is meshable or an occluder or capable of any orientation. First a file is selected by choosing the “Search” button. This button opens a file dialog which can be used to locate the imported file.

When a file has been selected, the scale factor for input must be taken into account. It is assumed that the objects have been defined elsewhere in their own coordinate system using a standard system of units. It is to be placed in a coordinate system based on the surface coordinates (usually centimeters). By selecting the correct radio button in Input for the Input Object and in Output for the Scene, a correctly sized object can be imported into the scene.

It is also possible to specify the initial position of the object in the scene with the three inputs in the “Position” section of the dialog. This is handy if a specified position is already known for the object, although the orientation may need to be fixed in SceneGen. If the initial position is not known, these coordinates can be left at zero and the object found later.

The Base Material Section allows the user to specify what the base material for the input object is on import. These values can also be set after the fact in the Edit Menu.

Some models are not defined in the same coordinate system used by SceneGen, with a right handed coordinate system with up being in the positive Z direction. For models that use a right handed coordinate system with positive Y being up, choosing the “Convert from Y to Z as up” rotates the object to comply with the SceneGen standard.



**Figure 13. The Import Object Dialog**

## The Visualization Area

Most of the control for SceneGen occurs in the Visualization Area. There are different modes that affect the different things you can do in the Visualization Area.

### Visual Modes



At the lower left corner of the Visual Editor section is the menu to select what visual mode that SceneGen is currently operating under. The current mode determines how the environment is viewed as well as what operations can be performed on the view and the items in the environment.

There are four possible modes available to the user.

"Perspective" mode is the starting mode, and it allows the user free reign for viewing the environment, and some selection tools. Perspective navigation is discussed further in the next section of this user's guide.

The other three modes available to the user are the "Orthographic" modes. These modes display with a parallel projection, without perspective foreshortening. In these modes, selected items can be moved and rotated in the environment. How these orientations and placements are changed is discussed in the Ortho Controls section of this user's guide, but it should be noted that first axis specified on the option is always positive to the right and the second axis is always positive in an upward direction, and

moving an item happens in those directions. Rotation occurs about the axis transverse to the other axes.

In other words, if the "XY Ortho" option was selected, by clicking on it, then positive X would be to the right, positive Y would be up, and rotation would be about the Z axis.

Another thing to note is the type of item that is being moved. Oriented items cannot be rotated about any axis but the Z axis - *because the top must always point upwards*. They can be moved in any direction, but are fixed from rotating about the X or Y axis. Independent items do not have this restriction.

## Perspective Navigation



The Perspective Navigation tool has been designed to be intuitive, but as with all interfaces, intuitive is in the eye of the beholder. The tool has three main areas - the "reset" circle at the center, the movement mode option buttons, and the navigation buttons (drawn as triangles around the "reset" circle).

### Reset Circle

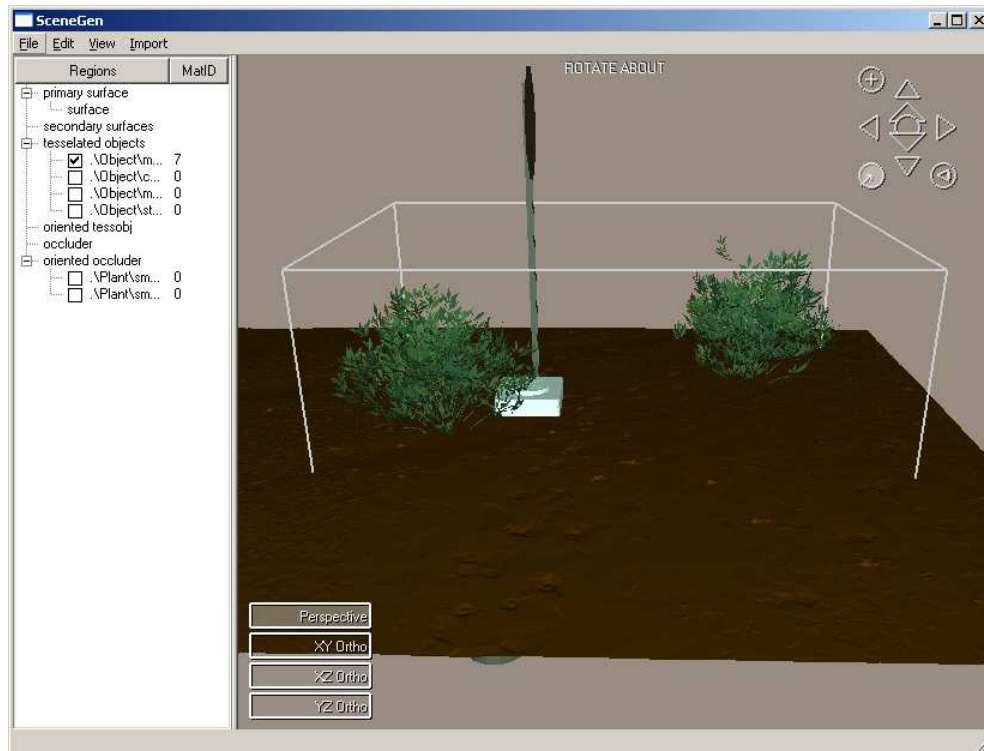
It is possible with SceneGen perspective navigation to become lost, and not know how to find your way back to a useful view of your data. This will be especially apparent as the different modes of navigation are discussed below. In order to keep the user from becoming frustrated, the reset circle was created. Clicking the mouse inside the "reset" circle resets the camera position to where it began at the beginning, looking directly down from above and with the positive X axis to the right. This also resets the center of rotation for the "rotate about" mode.

### Movement Mode

The movement mode option buttons are in the three circles surrounding the navigation controls. These modes affect how the camera is oriented, and how it is moved. There are three modes:



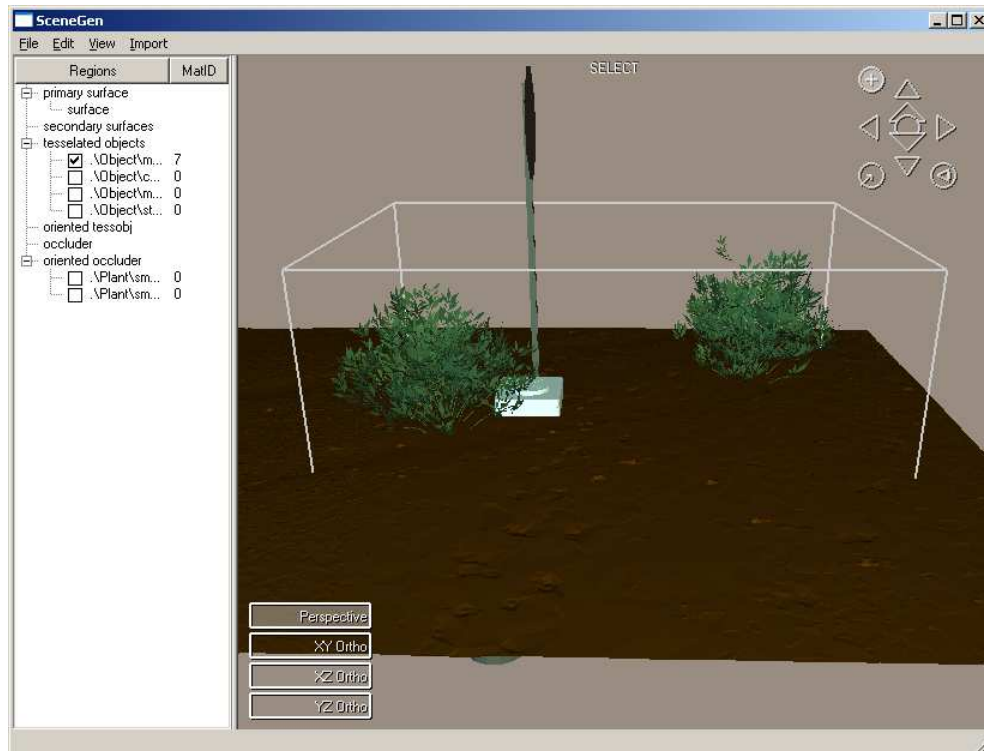
"Rotate About" is the default mode of orienting the camera to view the environment. It places the user's vantage point on a sphere centered about a point a fixed distance from the viewer. This fixed distance is set as the distance from the initial viewer position to the center of the environment. Click-dragging the mouse causes the viewer position to change about the data the same way that rolling a ball under your hand will change the contact point of your hand to the ball (try it and see). This is the default mode because it tends to keep the environment in the area of focus, although using the navigation buttons can change this.



**Figure 14. Rotate About Mode**



"Select" mode is not actually an orientation mode, but a mode for selecting objects in an environment to modify. Because SceneGen was designed to be used with a mouse and still be cross-platform compatible, there were no guarantees about the number of mouse buttons available to the user. For this reason, the interface was designed to work with a single mouse button but still function with multiple mouse buttons. This influenced the "select" mode. If the left button is clicked and released when in the Visual Editor section and over a selectable object, the object is marked selected. If the mouse has a "right" button, it behaves the same as if it were in the "Rotate About" mode.

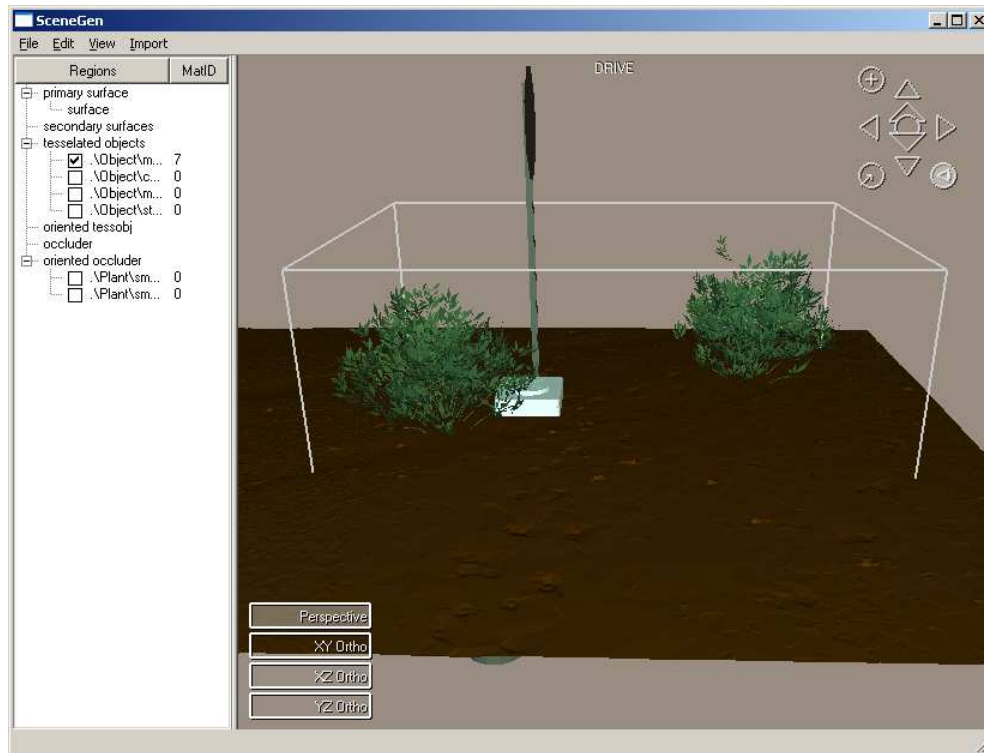


**Figure 15. Select Mode**



"Drive" mode is perhaps the easiest mode to understand. In this mode, the viewer position is fixed, but the direction and orientation of the view is changed by click-dragging the mouse. For example, if you want to look up, click-drag the mouse in the upward direction.





**Figure 16. Drive Mode**

## Navigation Buttons

The Navigation buttons are arranged like arrowheads to show the direction of panning that they allow. The top and bottom buttons move the point of view upwards and downwards, respectively, the left and right buttons pans the point of view to the left and right, and the buttons that overlap the reset circle pan forward and backwards. To pan, simply click and hold the mouse when over the appropriate button. The viewpoint will move slowly at first and accelerate until it reaches a top velocity. This allows the user some precision of movement.

NOTE: In "rotate about" mode, the point of rotation moves with the point of view. This means that the viewpoint will not rotate about the center of the area of interest for an environment.

## Orthographic Navigation



The orthographic menu overlay is similar to the perspective menu overlay, but it has no navigation elements except zoom features currently. It is composed of modes on the left and operations on the right. By selecting the appropriate mode, changes can be made to the orientation and position of objects, but only in the orthographic plane selected. The operation buttons allow useful operations that only need to occur once when the button is pressed.

## Zoom Mode



Zoom mode allows the user to drag out a region for zooming into. While Zoom mode is enabled, click-dragging with the left mouse button will show a highlighted region to be zoomed. The extents of the zoomed region will be expanded to fit the aspect of the visualization area. Multiple zooms will zoom into smaller and smaller regions.

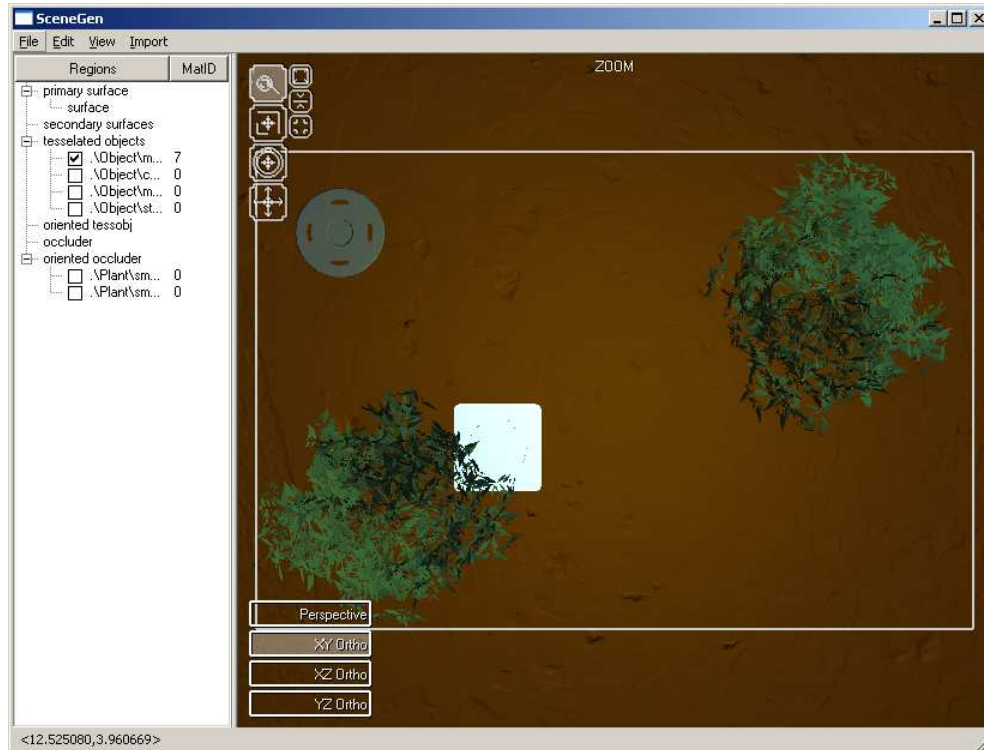
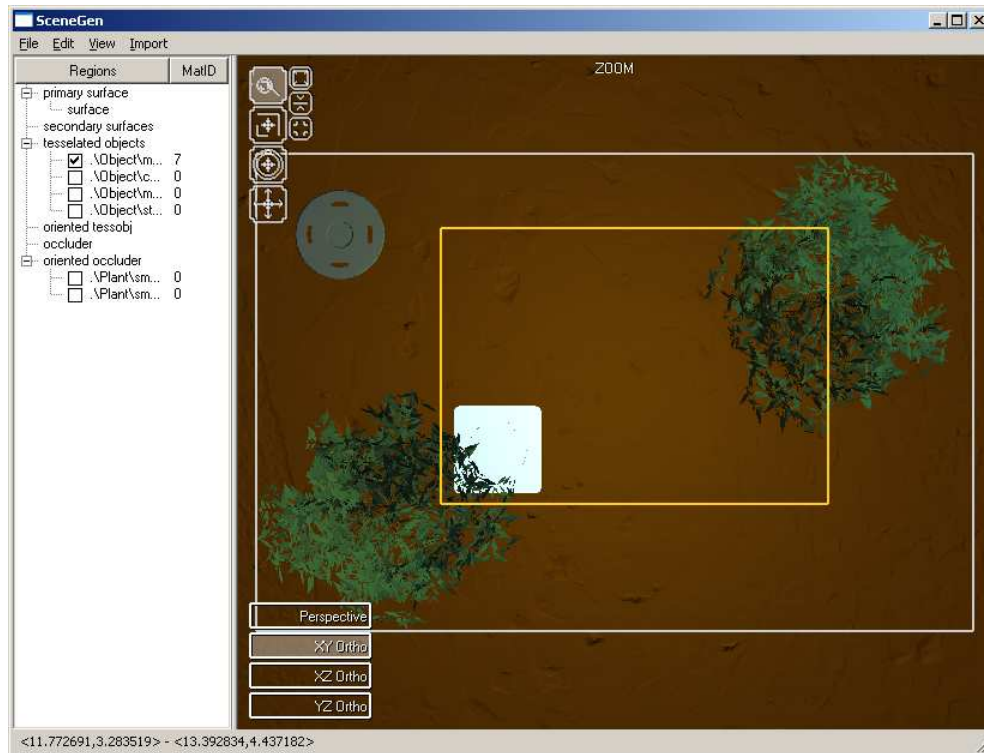


Figure 17. Zoom Mode - before zoom



**Figure 18. Zoom Mode - selecting the zoom region**

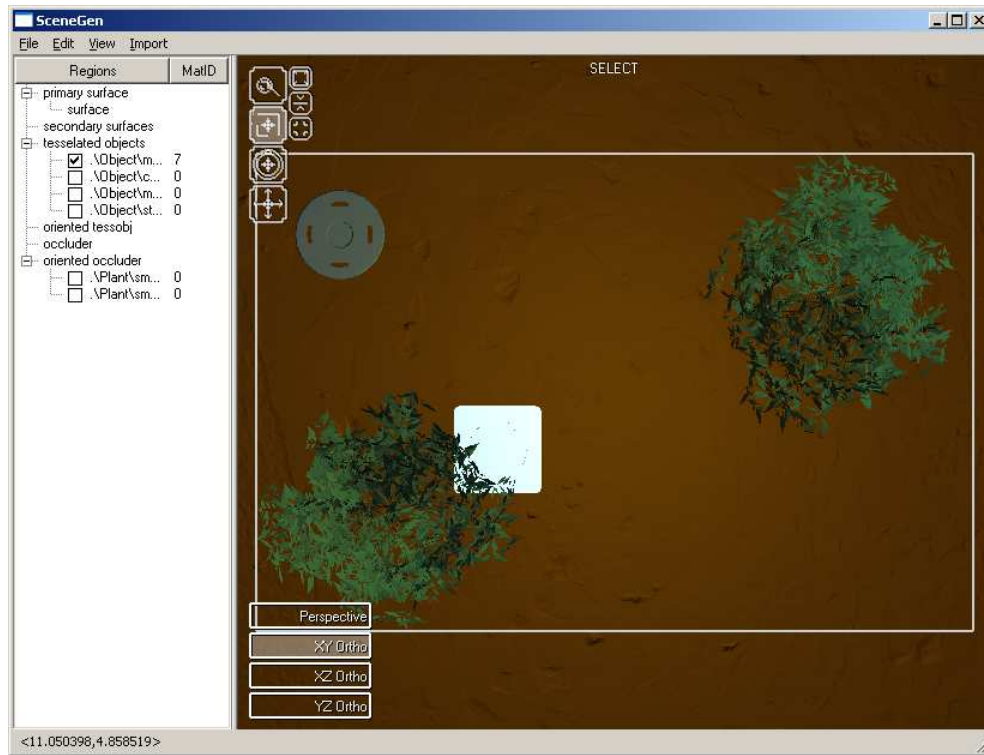


**Figure 19. Zoom Mode – zoomed**

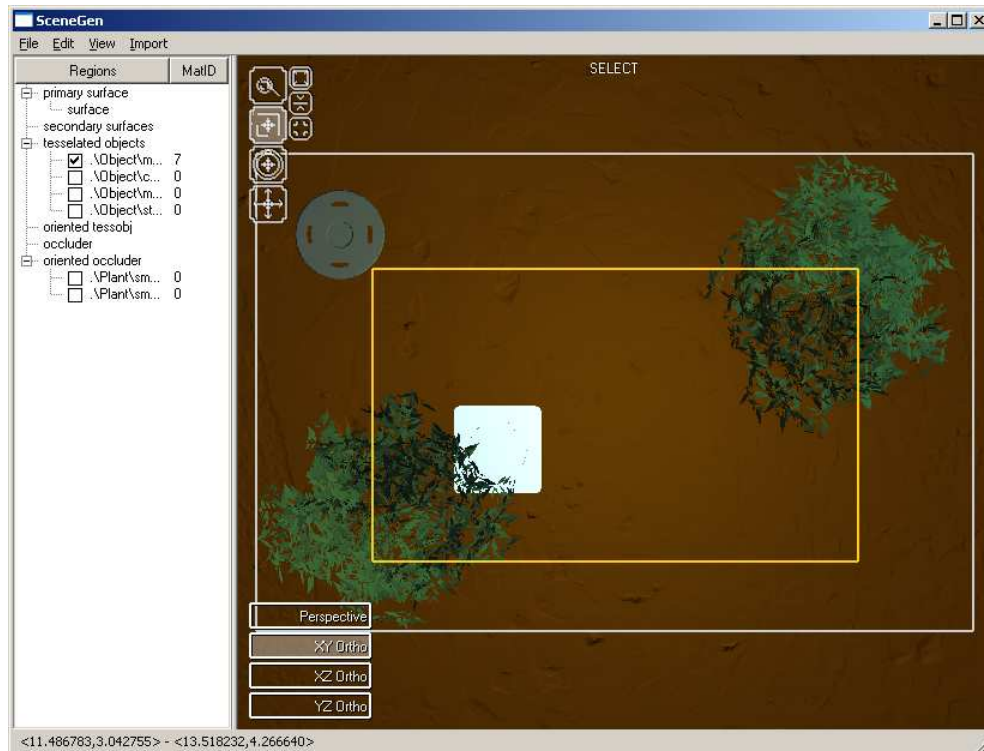
**Select Mode**



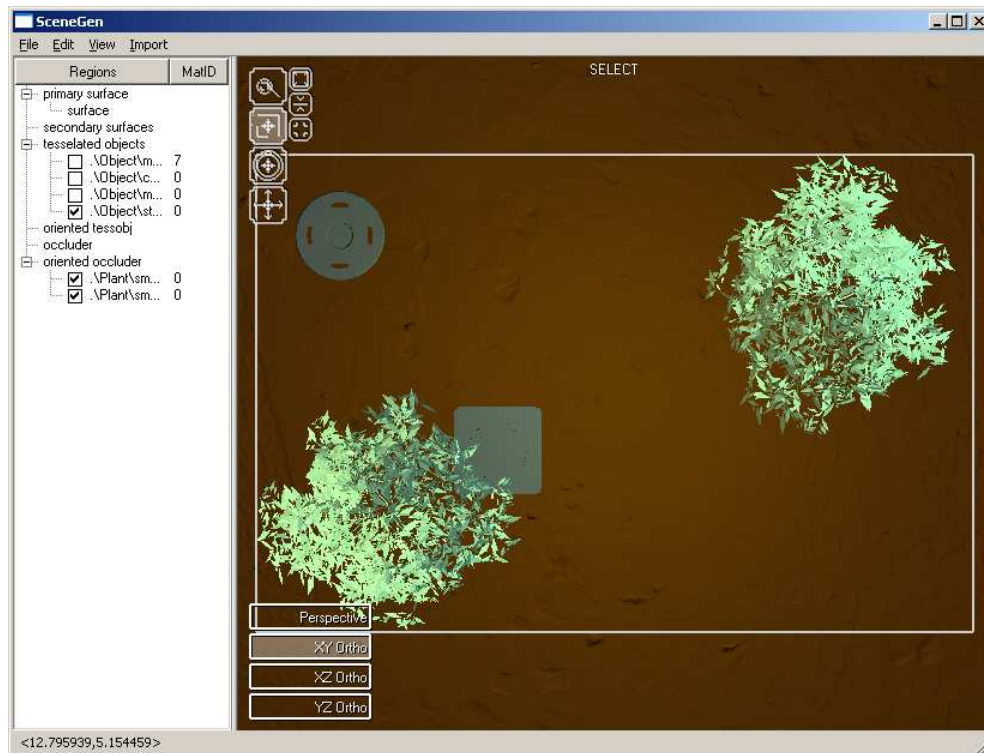
Select mode operates quite the same as Zoom mode except that, instead of zooming, objects are selected with the click-drag operation. In this way, multiple objects can be selected at once. The only objects that cannot be chosen this way are surfaces. Objects in the click-dragged region are toggled from selected to deselected and vice versa.



**Figure 20. Select Mode**



**Figure 21. Select Mode - selected area**



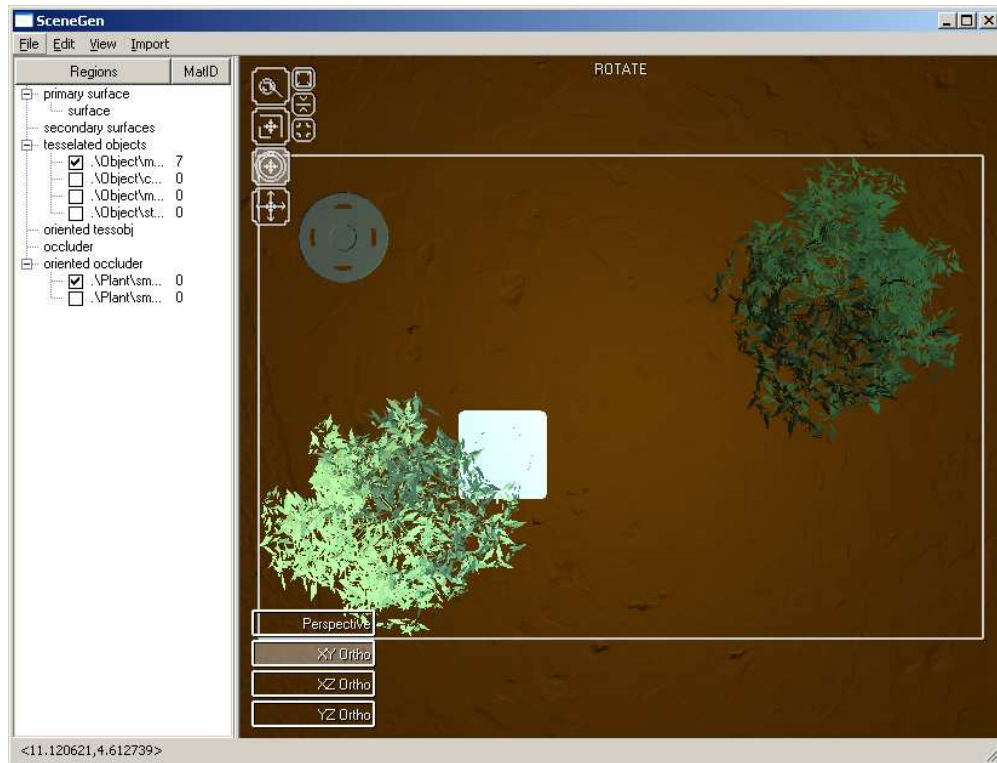
**Figure 22. Select Mode - objects toggled**

## Rotate Mode





In Rotate Mode, the user can rotate selected objects about a specified axis. The axis is specified when the user presses the left-mouse button, and is active until the user releases the button. The axis is transverse to the viewing plane (for example, in XY Ortho mode, the selected axis is along the Z coordinate), and items are rotated about the axis relative to their original position and orientation prior to the button press. The angle of rotation follows the position of the mouse to the axis, with zero degrees being directly up in the Viewing Area. A compass is displayed at the current axis and the degrees of rotation are displayed at the bottom of the screen. Note that rotating an object can affect its position as well. If the mouse is clicked on an unselected object, then it is selected and rotated also, and deselected after the mouse is released.



**Figure 23. Rotate Mode**

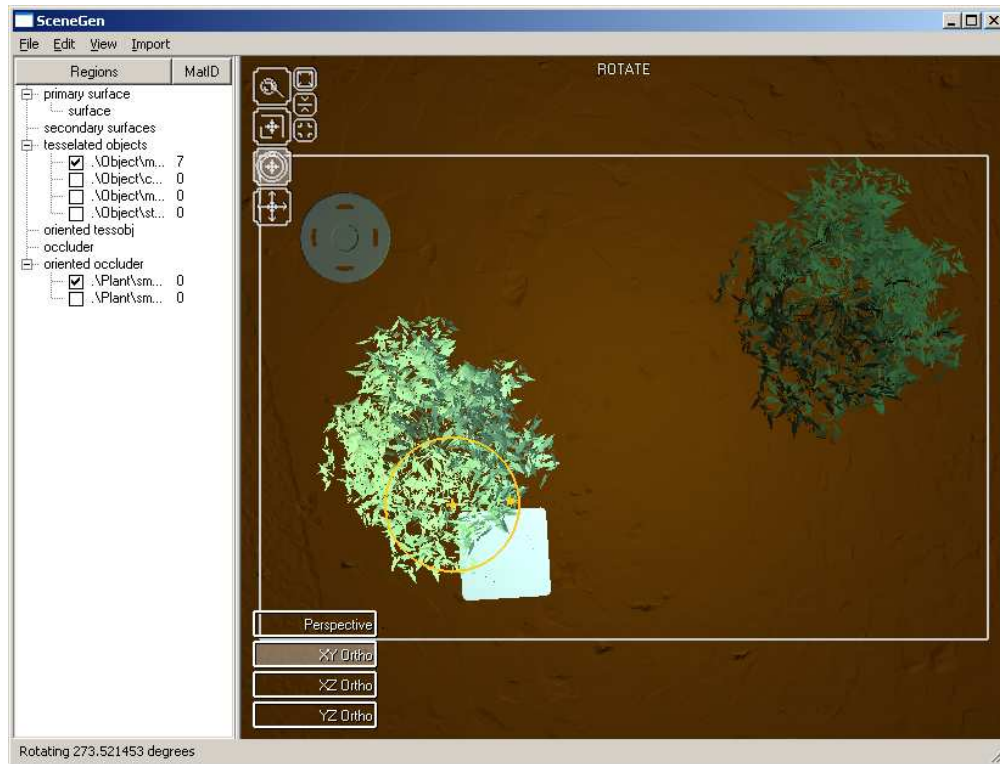


Figure 24. Rotate Mode - rotating

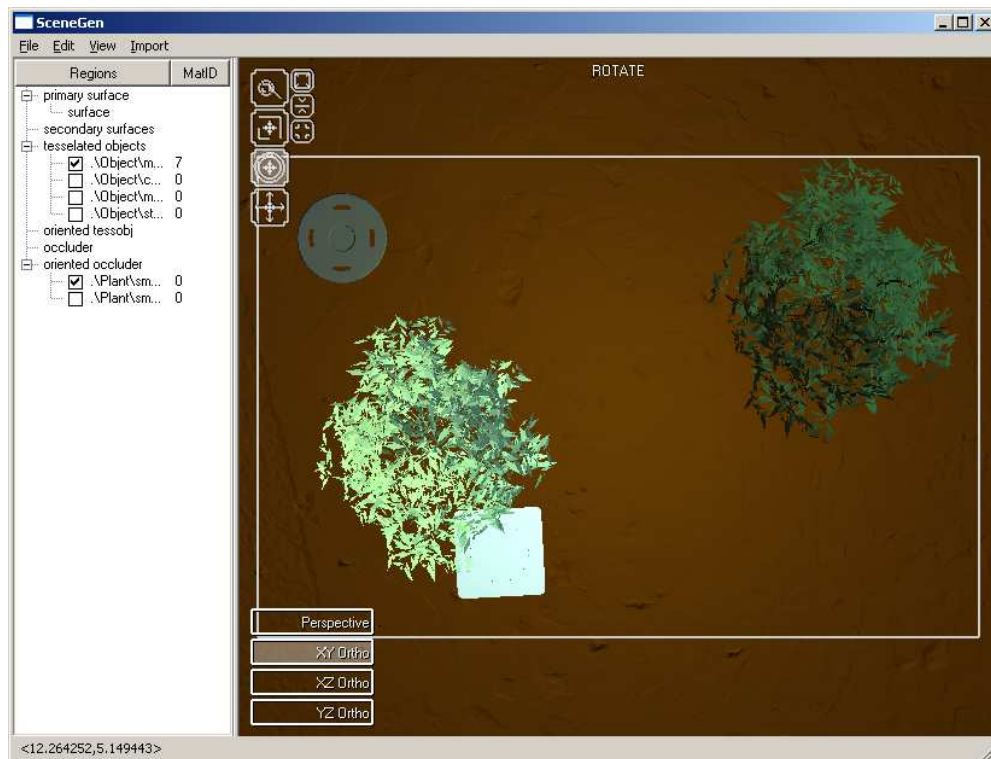
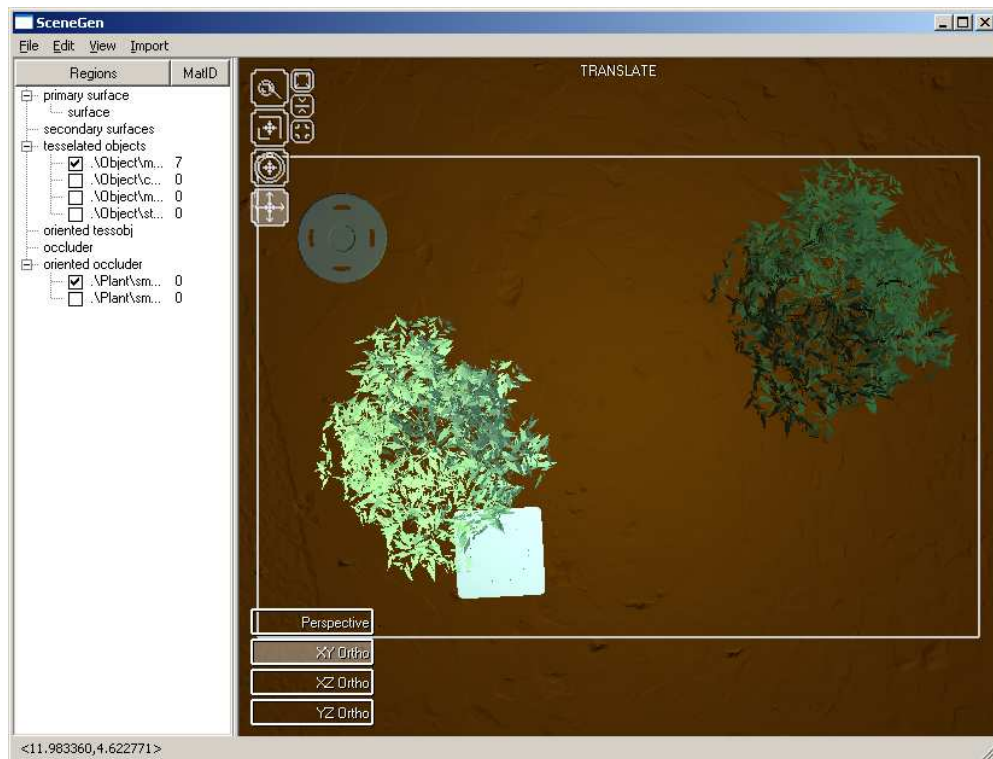


Figure 25. Rotate Mode - finished position and orientation

## Translate Mode

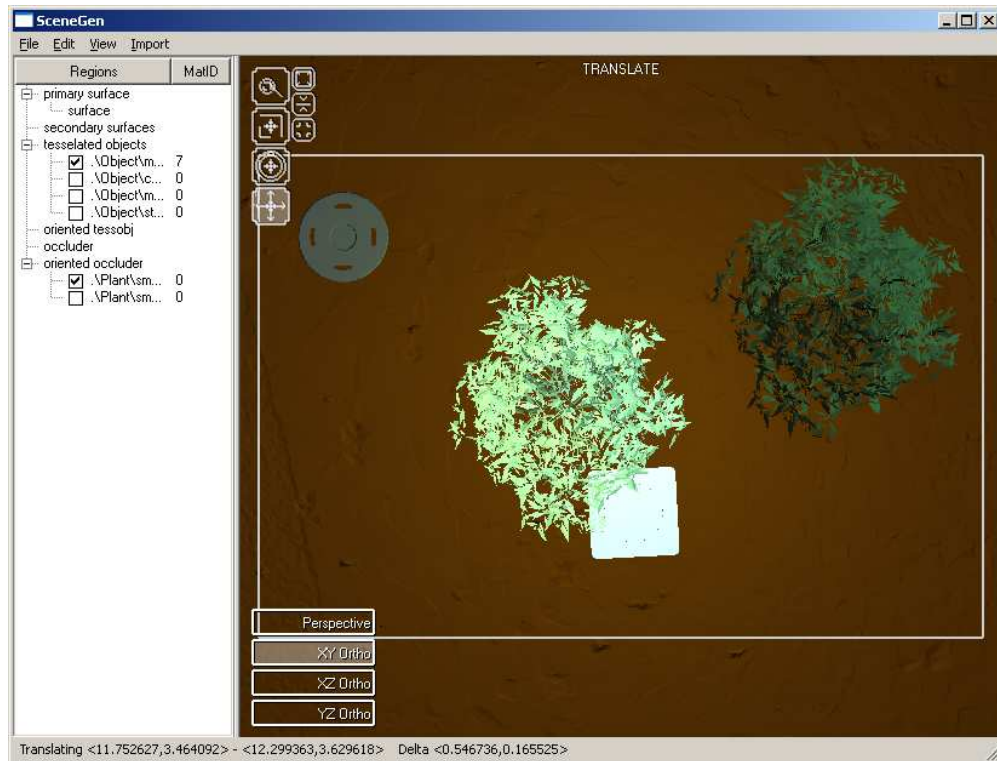


Translate Mode uses the click-drag method of moving selected objects. When the left mouse button is pressed, an offset point is established, and objects are offset by the same amount as the mouse is offset from the original point. The coordinates are displayed at the bottom of the screen. While translating, the display of coordinates shows not only the original location of the mouse but also the new location and the delta distance between them so the user can know how far objects are offset from their original position. Releasing the mouse locks the items at their new location. Orientation is not affected by this transformation. If the mouse is clicked on an unselected object, then it is selected and translated also, and deselected after the mouse is released.

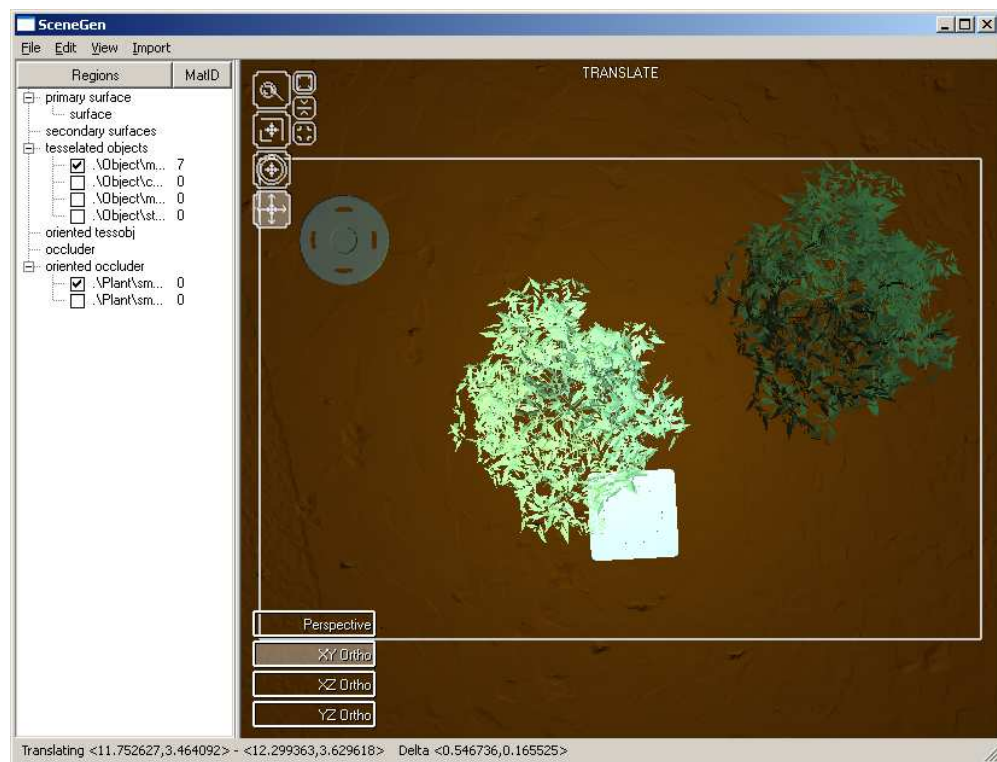


**Figure 26. Translate Mode**






**Figure 27. Translate Mode - moving objects**




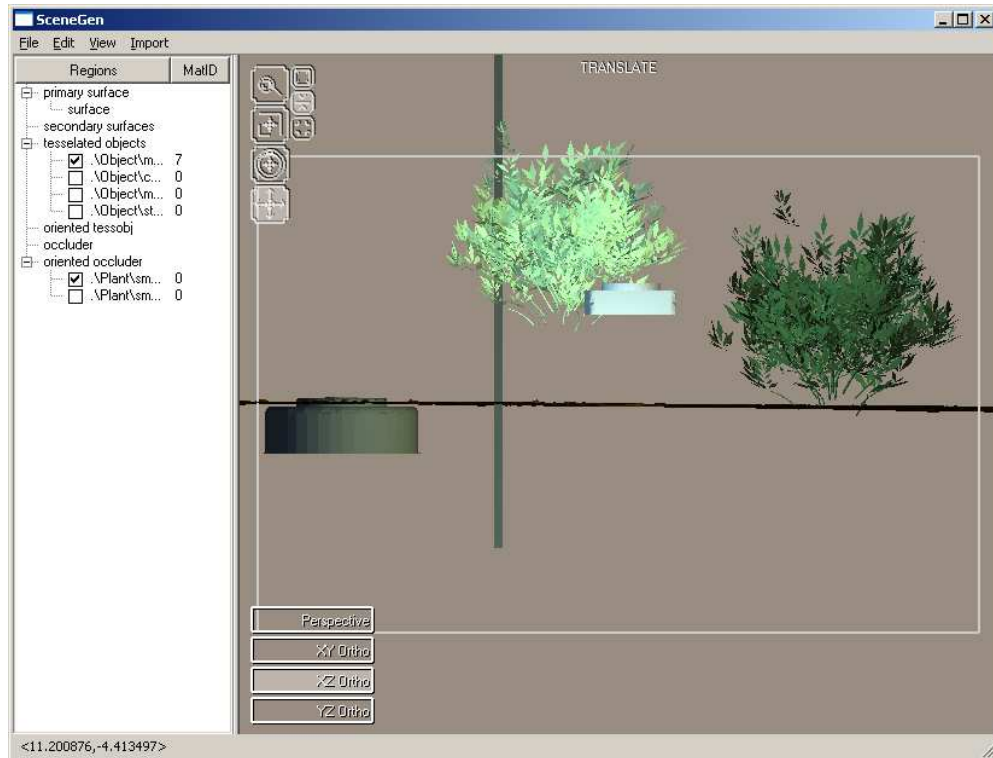
**Figure 28. Rotate Mode - Finished Position**

## Zoom Out Operation

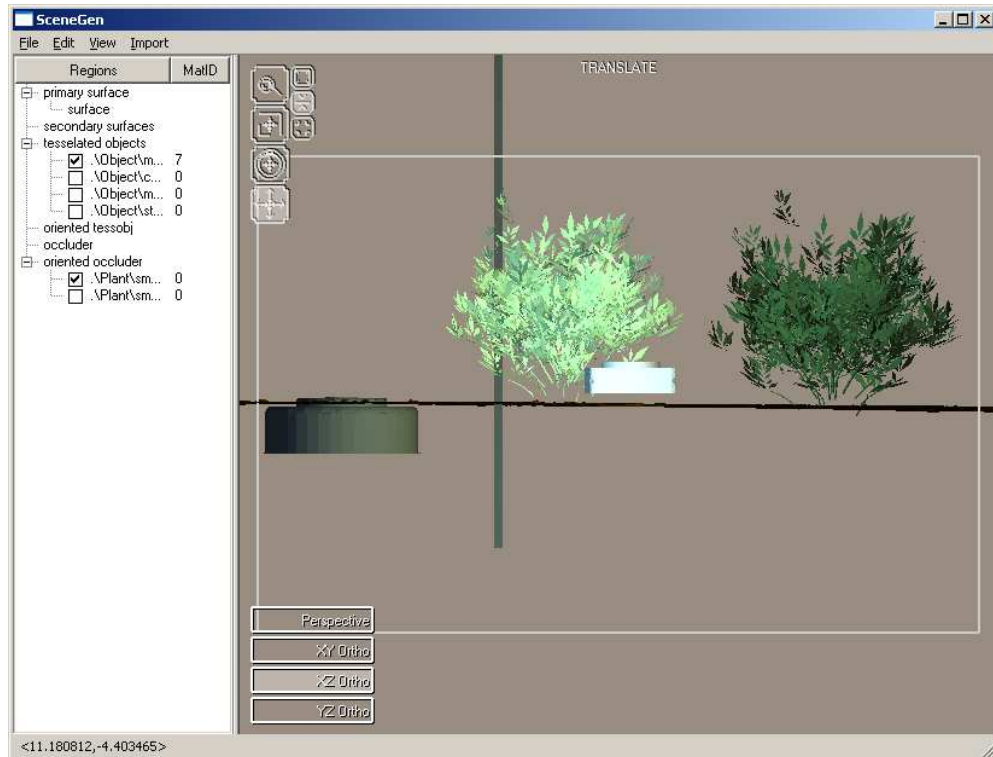
 Clicking this button causes the Orthographic view to be displayed at full extents, as specified by the user (i.e. surface extents, area of interest extents, etc.). In this way, the user can zoom out to view the data.

## Snap to Surface Operation

 Clicking this button projects any selected items to the primary surface. It does this by finding the closest point in the LiDAR or surface data to the current X and Y location of each object and setting the Z translation of the object to that point's Z coordinate. Notice that this is all relative to the origin of the selected object. If the origin of the object is the center of the object, the object will be embedded in the surface. If the origin is outside of the object, the object may not touch the surface at all.



**Figure 29. Before snapping to surface**



**Figure 30. After snapping to surface**

## Center Operation



This operation takes any selected objects and projects them to the center of the current extents. In this way, lost objects can be brought to an area where they can be selected and manipulated again.

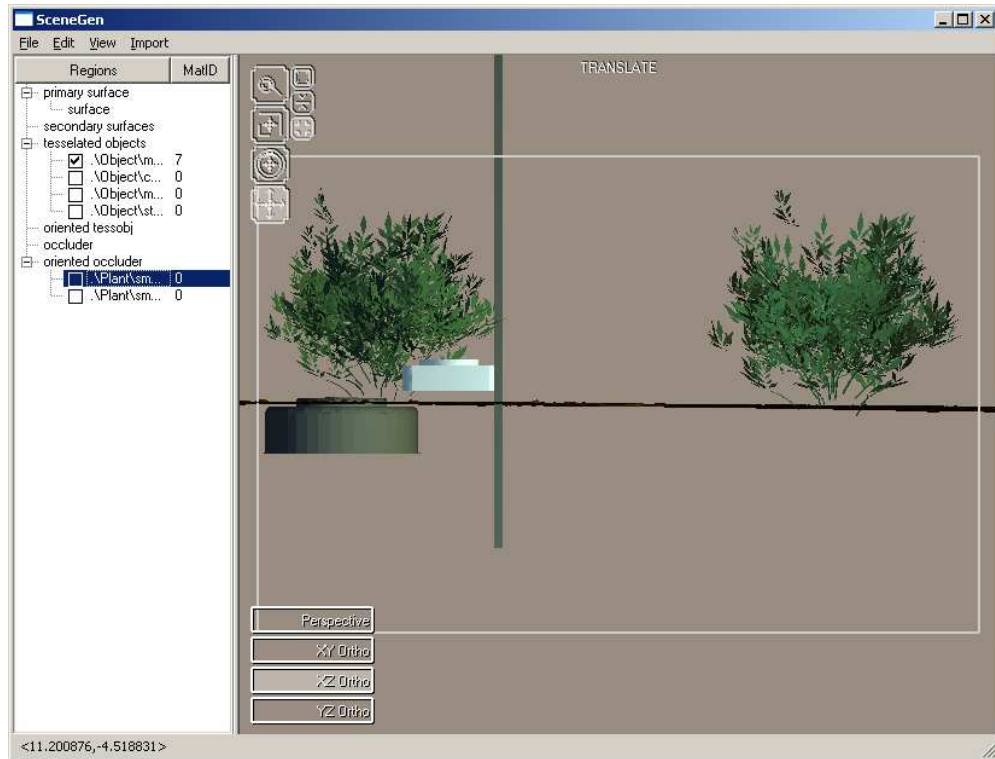


Figure 31. Before centering

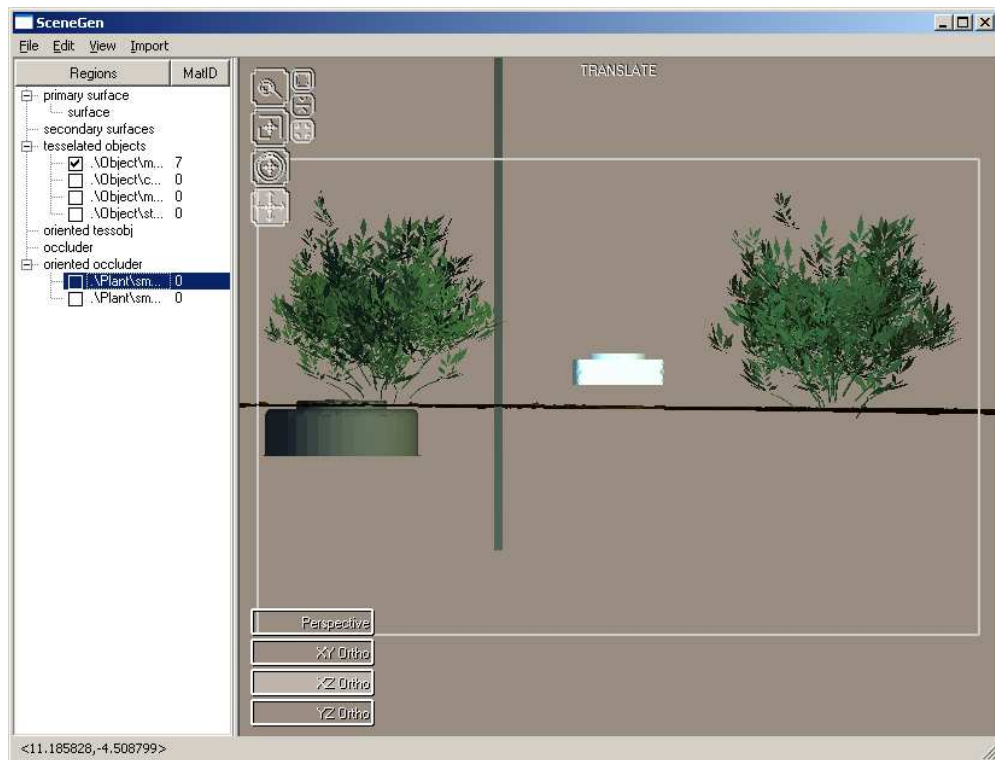


Figure 32. After the center operation

## A note on surface visualization

LiDAR data is stored as a set of points in space. Earlier versions of SceneGen attempted to visualize this data as a point set. There were several reasons why this was not desirable.

The first reason that this was not desirable was that the point sets coming from LiDAR could be very large. Data samples could be collected at 4mm resolution. Some of the data sets we were presented with had up to 4 million points, and larger data sets were planned.

The second reason for not dealing directly with the points was that the LiDAR data could be very noisy. Any minute obstruction would create a false surface point, from plants to animals. We were only interested in surface data, so these false data points would need to be averaged out.

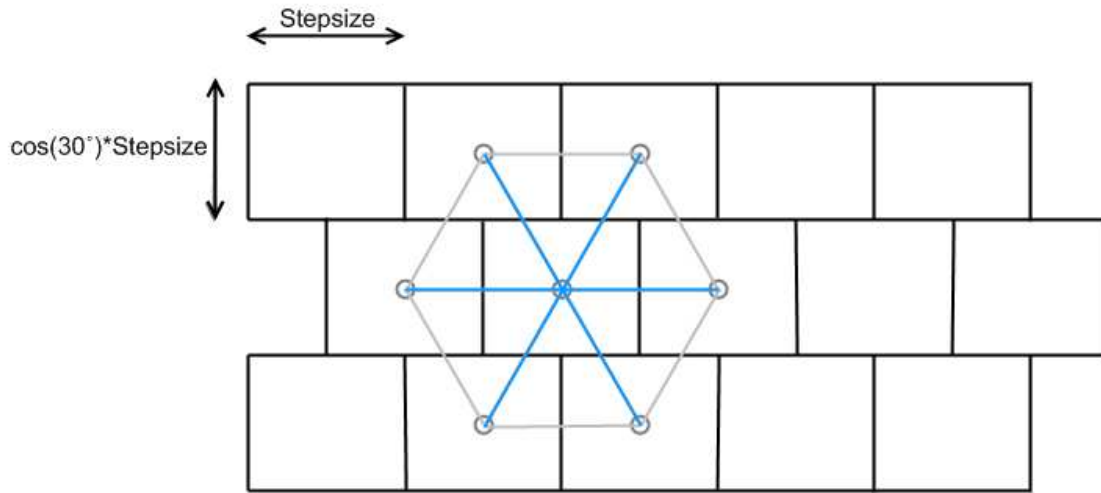
The third reason that a point set was not useful is that points by themselves do not reveal the curvature of the ground. For computer graphics, in order to light something, there must be a surface normal, and points do not have a surface normal. Lighting is important to distinguish the direction of a surface at a point on that surface.

Because of these three reasons, it was decided that we needed to implement an algorithm to determine a heightfield surface from the LiDAR data. The heightfield surface would need to be based local averages of sampled data at any point, so that noise could be minimized. The surface also needed to be simplified, using the least amount of triangles possible for fast visualization. Finally, the surface needed to maintain areas of high curvature, so that pertinent details could be maintained.

## Surface Definition and Simplification

The first step in surface definition is to attempt to create a surface from the cloud of points provided by the undeformed or deformed LiDAR data. The surface scans that we have dealt with have been high resolution ( $<1\text{cm}$ ), low altitude LiDAR scans of ground surfaces. See Figure 5(a). As such, there was random noise caused by ground cover such as vegetation. We need to represent this data as a ground surface that could be described with a mathematical function, i.e. a height field. In order to remove ground clutter, we re-sample the data with a specified distance between samples that would capture the relevant details that could impact our solution. Despite being used for height fields, this work could be extended using patches for manifolds.

In order to minimize the sampling memory footprint, in accordance with McGuire,<sup>5</sup> the samples are taken on a grid. Each adjacent row is offset by half the step size from the previous row and the step size is scaled by the cosine of 30 degrees in the row dimension to maintain an isometric grid, which would provide quality meshing before simplification of the surface. Sampling with this grid meant that we only needed to store a two dimensional array with Z values, and the X and Y values could be derived from the offset of the grid cell.



**Figure 33. An isometric grid from 2D array**

The heights are sampled by taking any LiDAR values within the step size radius cylinder from the center of the grid cell and averaging their heights. At present, these values are weighted equally, though future work may see a distance weighting applied. By averaging the values, we minimize the effect of ground clutter and restore the properties of a height field to our data. For grid cells that have no LiDAR data within their cylinder of influence, either due to shadow or some other reason, we apply a flag value (typically INFINITY) and deal with these values in the next step of the process.

At this point we can create a mesh from the grid, with each cell center being connected to six other cell centers, except on the outside edges. See Figures 5(b) and (c). Grid cells that have no data would be represented as discontinuities. From this mesh, we can simplify our data.

There are several methods for simplifying mesh geometry,<sup>6</sup> but given the hexagonal gridded nature of our data, a point removal scheme seemed warranted. After the points were removed, the surface could be reconstructed from the remaining points. The goals for removing points would be to eliminate points with low curvature. A measure of curvature at any point could be established by comparing the differences between a point and the three sets of lines between neighbors that pass through that point. Of course, this holds true only for height-field data.

In order to keep track of points that were to be kept and those which were to be removed, a “distance grid” was formed. This distance grid would hold the maximum distance of a point from the line connecting its neighbors or infinity for points that must be kept or zero for points that must be removed.

For each grid cell, the following procedure was performed. If the elevation of the grid cell was INFINITY, then the cell was flagged for removal in the distance grid. Then, each of the grid cells neighbors is checked. If any neighbor does not exist, then the current grid cell is flagged to be kept. In this manner, the external borders and the borders of any internal holes can be kept.

If a grid cell has not been flagged by the other two methods, then its maximum curvature along the three opposing sets of grid cells is measured and stored in the



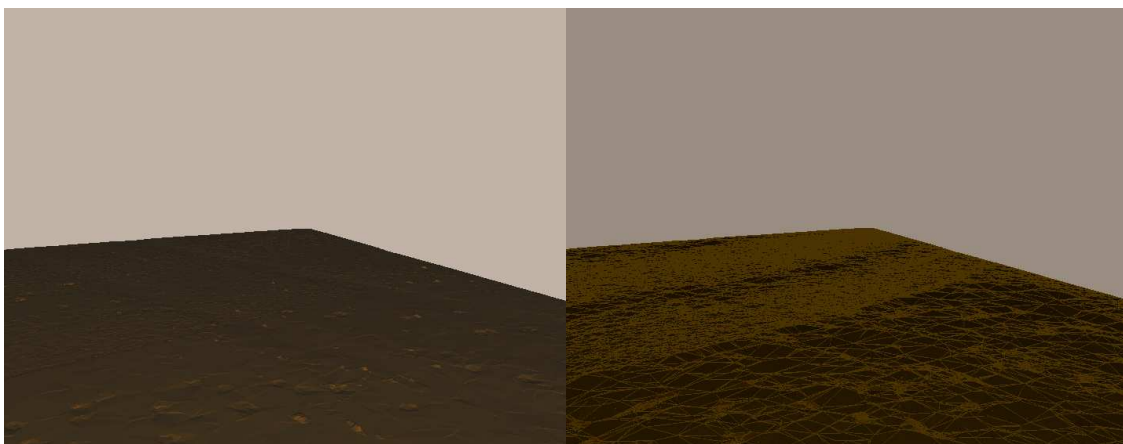
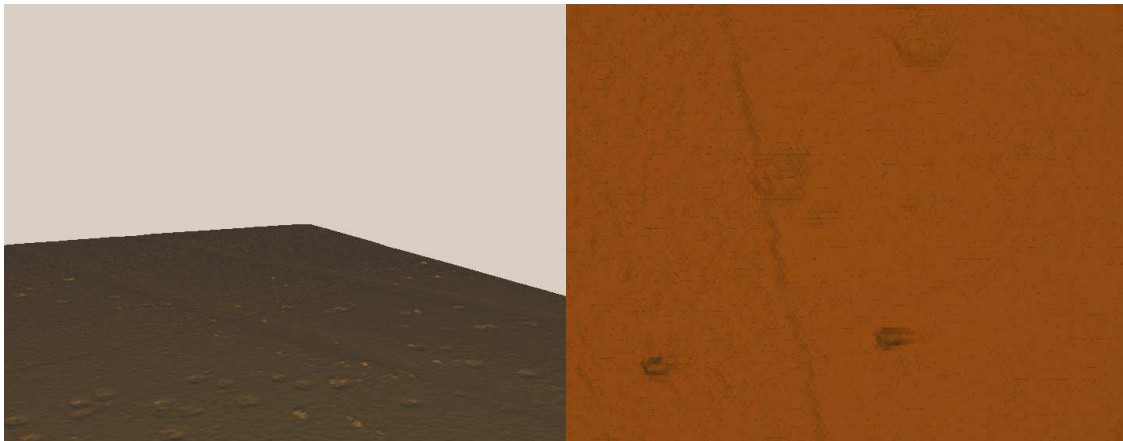
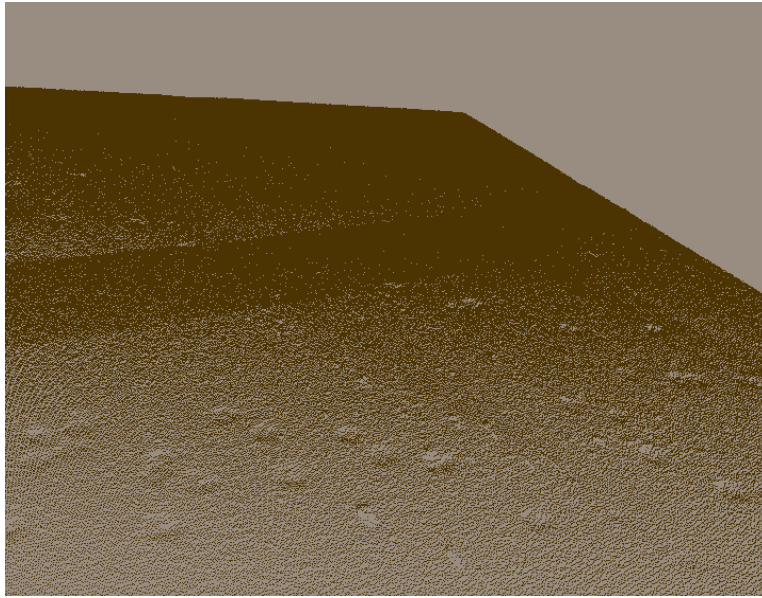
distance grid. In order to measure the curvature of the current grid cell against a set of opposing neighbors, the line segment between the opposing neighbors is computed and then the shortest distance from the current grid cell location to the line segment is found. This test is also performed for the other two sets of opposing neighbor points. The maximum of the three results is then stored in the distance grid.

After this procedure has been performed for each grid cell in the array, the points are culled from the array. The culling is based on a specified cut off value. Grid cells with corresponding values in the distance array that are less than the cut off value are culled. The un-culled points are stored for triangulation of the surface.

For our purposes, we set the cut off value based on the sine of a specified angle and the step size distance between points. More points will be removed with greater angles. There is a possibility that gradual curvatures could result in larger errors over long regions, but varying the cutoff angle can remedy these situations.

After we have a final set of high curvature points, we project these points onto the XY plane. These projected points are then run through a constrained Delaunay triangulation algorithm (in our case Johnathan Shewchuk's Triangle code <sup>7</sup>) to provide a surface triangulation. The two dimensional points are then mapped back to the original three dimensional point set to create the surface TIN (triangulated irregular network). See Figures 5(d) and (e).

Some consideration was given to other mesh simplification techniques. Most of these techniques were based on edge removal. When the edge is removed, its nodes must be collapsed to a single point and choosing that point can have a bearing on the quality of the resulting mesh.<sup>8</sup> This can have a large influence in mesh shape for our sampled data. There are also many different quadric error measures that could be applied to determine edge removal. Some advantages of these other mesh simplification techniques is their ability to deal with more than just TINS, and using other error measures than just curvature. For future work, we intend to explore these other methods.



<sup>5</sup>McGuire, M. and Sibley, P. G., “A Heightfield on an Isocetric Grid,” 2004.

<sup>6</sup>Heckbert, P. S. and Garland, M., “Survey of Polygonal Surface Simplification Algorithms,” Tech. rep., to appear.



- <sup>7</sup>Shewchuk, J. R., “Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator,” *Applied Computational Geometry: Towards Geometric Engineering*, edited by M. C. Lin and D. Manocha, Vol. 1148 of *Lecture Notes in Computer Science*, Springer-Verlag, May 1996, pp. 203–222, From the First ACM Workshop on Applied Computational Geometry.
- <sup>8</sup>Garland, M. and Heckbert, P. S., “Surface Simplification Using Quadric Error Metrics,” *Computer Graphics*, Vol. 31, No. Annual Conference Series, 1997, pp. 209–216.