

# **ParaView User's Guide (Version 1.6)**



# Table of Contents

1	Introduction .....	1
1.1	System Overview .....	1
1.1.1	The Menu Bar .....	2
1.1.2	The Toolbar .....	3
1.1.3	The Left Panel .....	3
1.1.4	The Display Area .....	3
1.1.5	The Status Bar .....	4
1.2	Getting Started.....	4
1.2.1	Installation.....	4
1.2.2	Starting ParaView .....	4
1.3	Terms and Concepts.....	5
1.4	A Simple Example.....	5
1.4.1	Start ParaView.....	5
1.4.2	Create a sphere .....	6
1.4.3	Change to wireframe.....	7
1.4.4	Change the resolution.....	7
1.4.5	Interact with the sphere.....	8
1.4.6	Add a bounding box.....	9
1.4.7	Shrink the sphere.....	9
1.4.8	Color by normals.....	10
1.5	Command Line Arguments .....	10
1.5.1	General .....	11
1.5.2	Client/Server .....	11
1.5.3	Parallel.....	12
1.5.4	Mesa.....	12
1.6	Running ParaView With MPI.....	12
1.6.1	Distributed Stand-Alone Mode .....	12
1.6.2	Client/Server Mode .....	12
1.6.3	Unix/Linux Related Issues .....	13
1.6.4	Windows Related Issues.....	13
1.7	Check for Updates .....	13
1.8	ParaView Support.....	13
2	Loading Data.....	14
2.1	Overview .....	14
2.2	Supported Data Formats.....	15
2.3	Data Types.....	16
2.4	ParaView Session Files .....	17
3	Sources.....	18
3.1	Overview .....	18
3.2	3D Text .....	19
3.3	Arrow .....	19
3.4	Axes.....	20
3.5	Box.....	21
3.6	Cone .....	22

## Table Of Contents

3.7	Cylinder.....	22
3.8	Line.....	23
3.9	Mandelbrot.....	24
3.10	Plane.....	25
3.11	Sphere.....	26
3.12	Superquadric.....	27
3.13	Wavelet.....	28
4	Filters.....	29
4.1	Overview.....	29
4.2	Calculator.....	30
4.3	Cell Centers.....	30
4.4	Cell Data to Point Data.....	30
4.5	Clean.....	30
4.6	Clip.....	30
4.7	Connectivity.....	31
4.8	Contour.....	31
4.9	Crop.....	31
4.10	Cut.....	31
4.11	Decimate.....	31
4.12	Elevation.....	32
4.13	Extract Edges.....	32
4.14	Extract Surface.....	32
4.15	Extract Grid.....	32
4.16	Feature Edges.....	32
4.17	Normals generation.....	33
4.18	Glyph.....	33
4.19	Linear Extrusion.....	33
4.20	Loop Subdivision.....	34
4.21	Mask Points.....	34
4.22	Outline.....	34
4.23	Outline Corners.....	34
4.24	Point Data to Cell Data.....	34
4.25	Probe.....	35
4.26	Quadric Clustering.....	35
4.27	Random Vectors.....	35
4.28	Reflection.....	36
4.29	Ribbon.....	36
4.30	Rotational Extrusion.....	36
4.31	Shrink.....	36
4.32	Smooth.....	36
4.33	Stream Tracer.....	37
4.34	Threshold.....	37
4.35	Triangle Strips.....	37
4.36	Subdivide.....	37
4.37	Tetrahedralize.....	37
4.38	Triangulate.....	38

4.39	Tube.....	38
4.40	Warp (scalar) .....	38
4.41	Warp (vector) .....	38
5	Selection and Navigation .....	38
5.1	Overview.....	38
5.2	Selection Window .....	39
5.3	Navigation Window .....	40
6	Toolbar.....	40
6.1	Overview.....	40
6.2	Camera Controls.....	41
6.3	Toolbar Filters.....	41
6.4	Center Of Rotation Controls.....	43
7	3D View Properties .....	43
7.1	Overview.....	43
7.2	Background Color .....	44
7.3	Advanced Render Parameters.....	44
7.4	LOD Parameters .....	45
7.5	3D Interface Settings .....	46
7.6	Corner Annotation.....	47
7.7	Orientation Axes .....	47
7.8	Standard Views.....	48
7.9	Stored Camera Positions .....	48
7.10	2D / 3D Camera Controls.....	48
7.11	Camera Control.....	50
8	Display Properties.....	50
8.1	Overview.....	50
8.2	View Settings .....	50
8.3	Color Options.....	51
8.4	Display Style .....	53
8.5	Actor Control.....	53
9	Data Information.....	54
9.1	Overview.....	54
9.2	Statistics.....	54
9.3	Bounds.....	55
9.4	Extents.....	55
10	Saving Results.....	55
10.1	Overview.....	55
10.2	Supported Data Formats.....	55
10.3	ParaView Session Files .....	55
10.4	Supported Image Formats .....	56
11	Copy and Print.....	56
11.1	Overview.....	56
11.2	Image Copy.....	56
11.3	Image Print.....	56
12	New features .....	56
Appendix A:	3D Widgets.....	57

## Table Of Contents

Appendix B: Tutorials .....	58
Tutorial I: Visualizing Streamlines .....	58
1. Load the Data File.....	58
2. Extract Sub-grids .....	60
3. Create Streamlines .....	61
4. Add Tubes.....	62
5. Save the Tubes.....	63
Tutorial II: Animating Isosurfaces .....	64
1. Load the Data File.....	64
2. Create an Isosurface.....	66
3. Clip the Isosurface .....	67
4. Animate the Isosurface .....	68
5. Delete the Contour and Clip Filters .....	69
6. Animate a Cut-Plane.....	70

# 1 Introduction

## 1.1 System Overview

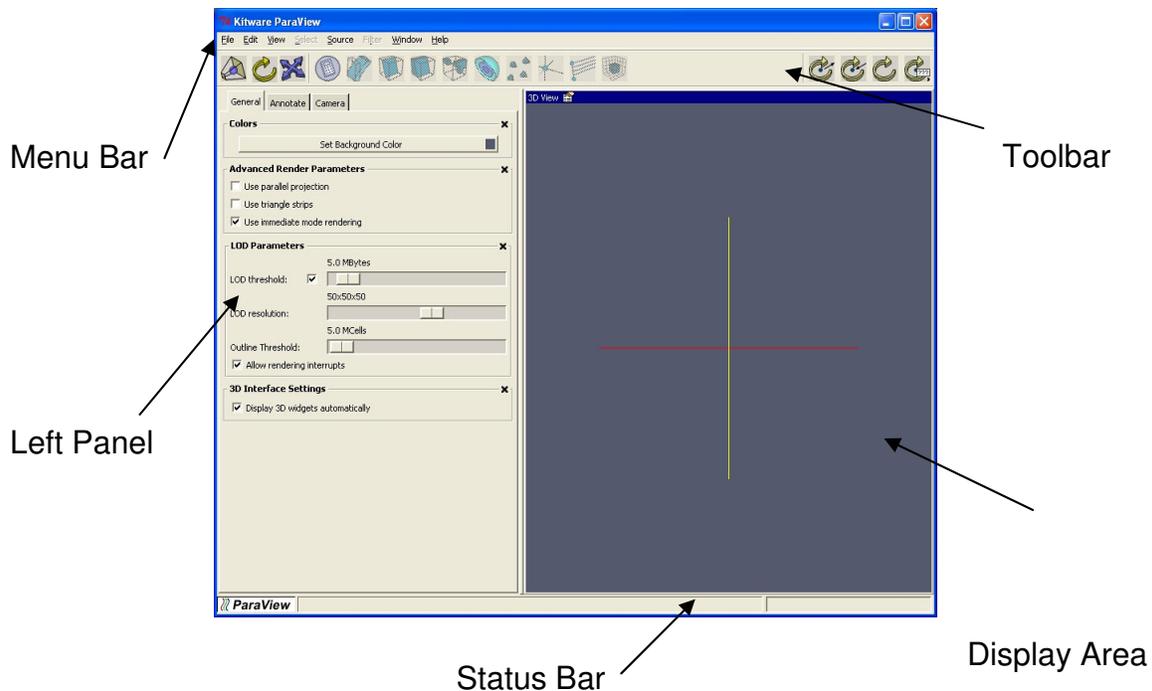
ParaView is a turnkey application for visualizing two- and three-dimensional data. It runs on computer platforms ranging from single processor workstations to distributed memory supercomputers. This allows ParaView to scale from a stand-alone single-process application to a multi-process system handling large data sets by distributing the data across multiple processors. The goals of the ParaView project include the following:

- Develop an open-source, multi-platform visualization application.
- Support distributed computation models to process large data sets.
- Create an open, flexible, and intuitive user interface.
- Develop an extensible architecture based on open standards.

ParaView uses the Visualization Toolkit as the data processing and rendering engine and has a user interface written using a unique blend of Tcl/Tk and C++. This architecture makes ParaView a very strong and flexible visualization tool. Since all VTK data sources and data processing filters are either immediately accessible or can be added by writing simple configuration files, ParaView users have access to hundreds of state-of-the-art data processing and visualization algorithms. Furthermore, the use of the Tcl scripting language as a core component allows users and developers to modify the processing engine and the user interface of ParaView to suit their needs.

ParaView's user interface and operations are closely tied to the pipeline model in VTK. Loading data creates a VTK reader object, and processing data (e.g., creating a contour) adds a VTK filter to the pipeline. Creating a single non-branching pipeline using ParaView is trivial. Each new filter adds to the end of the pipeline. ParaView also allows advanced users to construct, navigate, and modify complex branching pipelines.

To learn more about ParaView, visit the web site (<http://www.paraview.org>).



**Figure 1. ParaView at startup**

At startup, ParaView will appear as shown above. There are several regions composing the user interface including the Menu Bar along the top of the application, the Toolbar just below the Menu Bar, the Left Panel on the left side which may contain a Property Sheet and the Selection / Navigation Window, the Display Area on the right side, and the Status Bar along the bottom edge. Each of these areas is described in more detail below.

### 1.1.1 The Menu Bar

The top menu bar provides menu buttons for loading and saving data, changing property sheets, creating sources and filters, viewing other windows, displaying help, and other standard functionality. Each menu is described in more detail below.

**File:** The File menu can be used to load and save data files, load and save session files, save and print images, and exit the application.

**Edit:** The Edit menu can be used to delete all data from the application, and to copy the current image to the clipboard. (Copy functionality is only available on Windows.)

**View:** The View menu can be used to select the Property Sheet displayed in the Left Panel of the application. Most of the functionality of ParaView will be contained in these property sheets. This is where the properties of sources and filters are set; application and 3D display settings such as background color, level of detail parameters, and camera controls are located; and animations can be created.

**Select:** The Select menu also controls the Property Sheet displayed in the Left Panel of the application. The Select menu contains an entry for each data object loaded or

created in the system. Selecting one of these will bring up its corresponding property sheet in the Left Panel area of the user interface. There are also a few standard glyph sources that can be accessed through the Glyphs submenu. Selecting a source using the Select menu is similar to selecting it using the Selection / Navigation Window.

**Source:** The Source menu contains a list of sources that can be created. After selecting a source from the list, the property sheet for that source will be displayed in the Property Sheet area. Set the parameters then press the Accept button to create the source, or the Delete button to remove the source from the pipeline.

**Filter:** The Filter menu contains a list of filters that can be applied to the currently selected data object. This will generally be the last data object created or the one most recently selected from the Select menu or the Selection / Navigation Window. The list of filters enabled in the Filter menu will change to correspond to the type of the selected data object. Once a filter is selected, the Property Sheet for that filter will be displayed. Set the parameters and press Accept to add the filter to the pipeline just after the currently selected data object, or press the Delete button to remove the filter from the pipeline.

**Window:** The Window menu can be used to hide or display the Left Panel. In addition, the Window menu can be used to bring up the Command Prompt, Error Log, or Timer Log windows.

**Help:** The Help menu can be used to display information about downloading this help document, to show version information for ParaView, or to run the ParaView demo.

### 1.1.2 The Toolbar

The Toolbar is located directly below the Menu Bar. The Toolbar contains buttons for resetting the camera, switching between 2D and 3D interaction modes, and changing the center of rotation. In addition, the Toolbar contains buttons for some common filters.

### 1.1.3 The Left Panel

The Left Panel displays the current Property Sheet on the bottom and the Selection / Navigation Window on the top. The Selection / Navigation window will not appear until data has been loaded or created. The Left Panel may be hidden or revealed using the Window menu on the Menu Bar. The current content of the Left Panel is controlled by the View and Select menus in the Menu Bar or by selecting a data object from the Selection / Navigation Window.

### 1.1.4 The Display Area

The Display Area is where the 3D representation of the scene is rendered. Mouse interaction is provided in this area.

## 1.1.5 The Status Bar

The Status Bar on the bottom edge of the application will provide short help messages, progress updates, and an error indicator (a small red exclamation point in the far right corner). Clicking on the error indicator will bring up the Error Log and change the color of the exclamation point to black.

## 1.2 Getting Started

### 1.2.1 Installation

#### 1.2.1.1 Unix Installation:

Uncompress the appropriate distribution (gunzip for .gz, uncompress for .Z), and change the current directory to where you want the files to be extracted and extract (tar xf). This will install the binaries into the bin/ directory, the documentation into the doc/paraview-1.6 directory, and support files into the share/paraview-1.6/ directory. We recommend installing in /usr/local or /opt.

#### 1.2.1.2 Windows Installation:

Run PVSetup.exe and follow the instructions. By default, this will install the ParaView executable and support files in the C:\Program Files\ParaView 1.6 directory.

### 1.2.2 Starting ParaView

There are a few different methods for starting ParaView depending on whether it is being run in single-processor stand-alone, distributed stand-alone, or client/server mode. The simplest ParaView execution mode is single-processor stand-alone:

#### 1.2.2.1 On Unix:

If it was installed in a directory that is currently in your search path (\$PATH or \$path), you can invoke ParaView from the command line with:

```
$ paraview
```

If this fails, you will have to either modify your search path or invoke ParaView by specifying the full path of the executable. For example, if the installation path is /usr/local/bin, the following will launch the application:

```
$ /usr/local/bin/paraview
```

ParaView recognizes many command line arguments. For a list of available arguments, invoke ParaView with the --help option. A complete list of command line arguments and their descriptions is contained later in this user's guide.

#### 1.2.2.2 On Windows:

Start ParaView by selecting "Programs | ParaView 1.6 | paraview" from the Start menu.

After starting ParaView, you can play the demo for a visual introduction to some of the capabilities of the application. To start the demo, select “Play Demo” from the Help menu.

## 1.3 Terms and Concepts

Because ParaView uses the Visualization Toolkit as the data processing engine under the hood, this document contains a few visualization concepts borrowed from VTK. The most important of these are:

**Object:** An abstraction that models the state and behavior of entities in a system.

**Data Object:** An object that is an abstraction of data. For example, when loaded, the contents of a Plot3D file is represented by a data object.

**Process Object:** A visualization object that is an abstraction of a process or algorithm. For example, the isosurfacing algorithm is implemented as a process object.

**Source:** A process object that produces at least one output (data object).

**Reader:** A source object that reads data from one or more data files

**Filter:** A process object that takes at least one input and generates one output. Filters include contour filter, cut-plane filter, streamline filter, sub-extent extraction filter, etc.

**Module:** In ParaView, process objects and the associated user interface for manipulating these objects are referred to as modules. ParaView modules include EnSight Reader module (reader), Sphere module (source), Stream Tracer module (filter) and ParaView Data Writer module (writer).

**Visualization Pipeline:** In ParaView (and in VTK), process objects are connected to each other to form a visualization pipeline. Each process object in the pipeline represents an operation performed on the data.

For a detailed explanation of these concepts, refer to *The Visualization Toolkit, An Object-Oriented Approach to 3D Graphics* by Will Schroeder et al.

## 1.4 A Simple Example

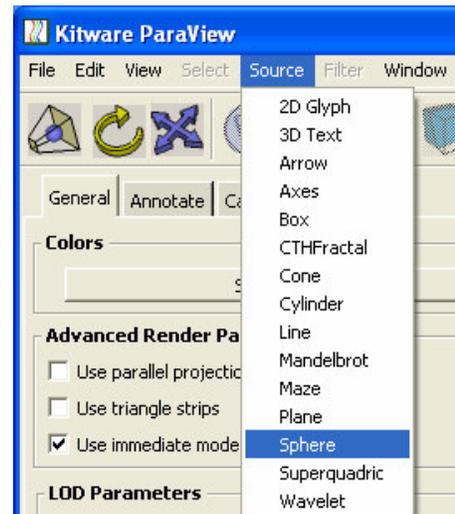
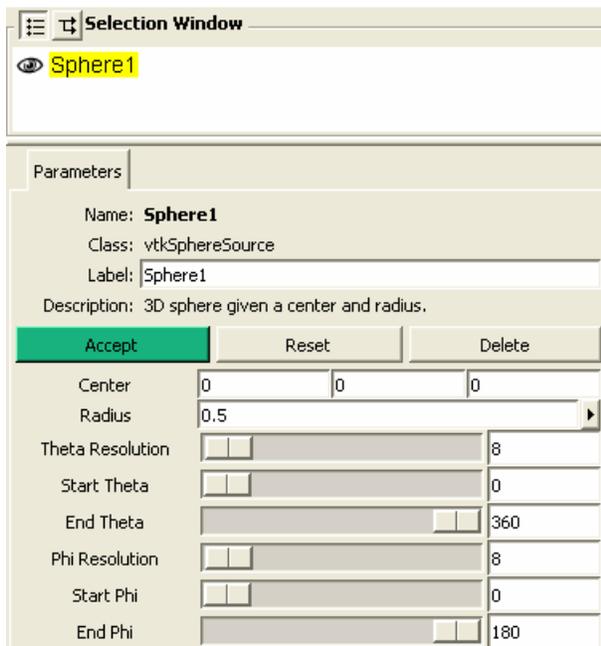
A simple ParaView example will be presented in this section in order to familiarize you with the basic operation of the application. Later in this User's Guide there are several detailed examples which should be attempted after you are comfortable with the basics.

### 1.4.1 Start ParaView

See section 1.2.2, Starting ParaView, for instructions on launching the ParaView application.

## 1.4.2 Create a sphere

ParaView starts with an empty scene. At this point you can either load data or create a source. In this example, we will create a sphere source using the Source menu in the Menu Bar as shown on the right. Once you select the Sphere item, the Left Panel will change to display the Selection Window on top and the Property Sheet for the sphere source on the bottom. As shown below, the Accept button on this property sheet will be highlighted in green, indicating that the data object created by this source is out of date. In this case it has not yet been created. Press the Accept button to create the sphere. You will see a white sphere appear in the Display.



For users who are familiar with VTK, here are the details of what has happened. A vtkSphereSource was created, and the parameters of this object are presented to you in the property sheet. When you first create the vtkSphereSource it has an empty output, and therefore you need to press the Accept button to cause an Update() call on the sphere source. This is also true when you change a parameter on the property sheet since the output will then be out of date with respect to the currently displayed parameters.

When you add data to ParaView (by loading data, creating a source, or filtering) a set of objects are created to manage the data generation, data storage, and display.

ParaView groups these objects together under the name presented in the property sheet - in this simple example it is Sphere1. When you select Sphere1 from the Select menu or by using the Selection / Navigation Window, the property sheet that is displayed contains parameters for the source object that creates the data, information about the data object that is created, and controls for the display of the data object. For example, the Parameters tab contains entry boxes for the center and radius of the sphere which are variables of the vtkSphereSource. The Information tab tells you that the data object created by the sphere source contains polygonal data with 96 cells and 50 points. The Display tab provides controls for the mapper, actor, and property objects associated with the display of this data, allowing you to change position, orientation, color modes and other display attributes.

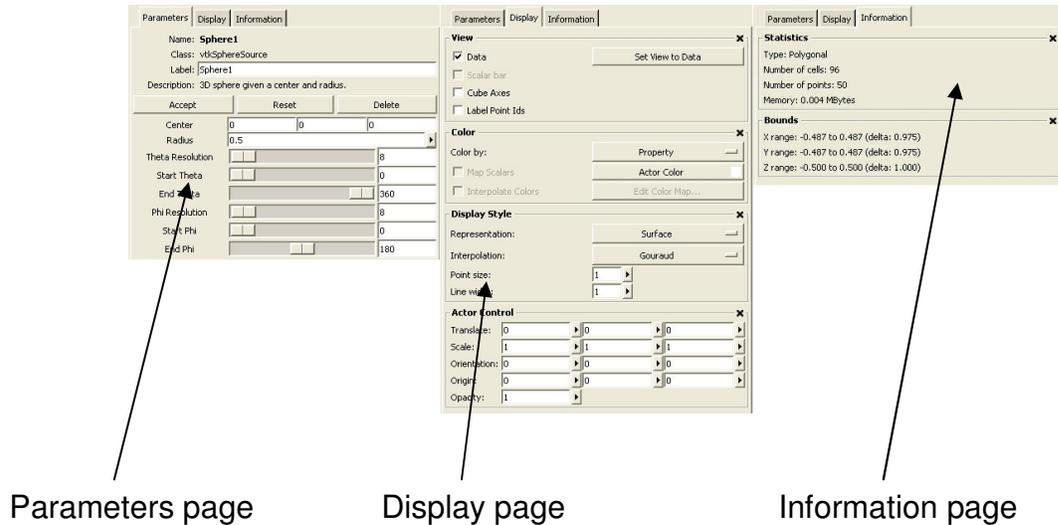


Figure 2. Module Property Sheet

### 1.4.3 Change to wireframe

The image you see in the Display Area should be similar to the one shown below on the left. We will now change the sphere from a solid representation to a wireframe representation to more easily see the underlying polygons. To do this, click the Display tab on the property sheet (shown in the Left Panel) for the sphere. In the Display Style area, change the Representation from Surface to Wireframe of Surface. The image should now appear similar to the one shown below on the right.

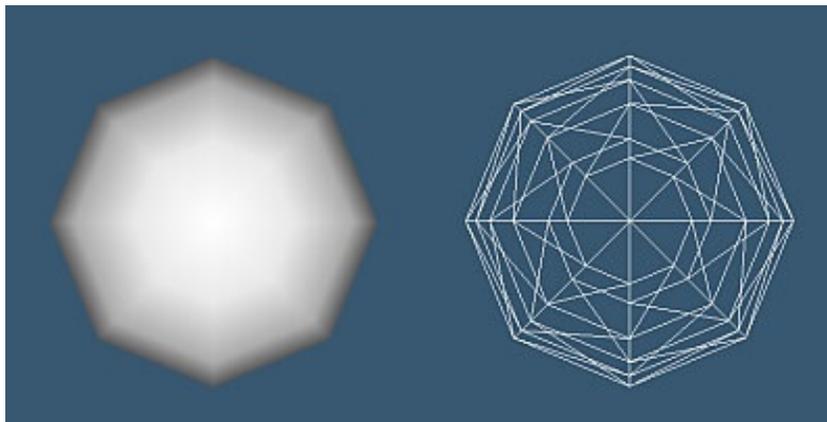
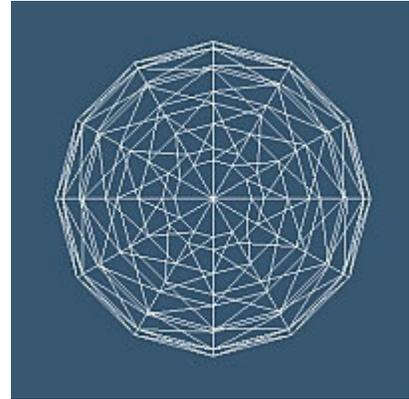


Figure 3. Surface vs. Wireframe

### 1.4.4 Change the resolution

We will now change the resolution of the sphere. Go to the Parameters tab on the property sheet shown in the Left Panel. There are two resolution parameters, Theta Resolution and Phi Resolution, each of which has a default value of 8. You can see

from the silhouette of the spheres shown above that there are eight segments defining the sphere. Change both of these resolution values to 12. You will see that the Accept button changes to green indicating that the displayed object is out-of-date with respect to the parameters in the user interface. Press the Accept button to see the result of this resolution change. You should see a sphere similar to the one shown on the right. This sphere contains more polygons that the previous one and provides a closer approximation of an actual sphere.



Note that changing the values and pressing the Accept button did not generate a new data object; it simply replaced the existing one. If you have changed some parameters and the Accept button is highlighted in green, but you decide that you would like to cancel this change, simply press the Reset button. This will restore the values previously used to generate the data (or the default values if the data has not yet been generated).

### 1.4.5 Interact with the sphere

Using various mouse and keyboard combinations, you can control the objects and the camera in the Display Area. The mouse bindings can be viewed and changed by selecting the 3D View Properties option from the View menu, then clicking on the Camera tab.

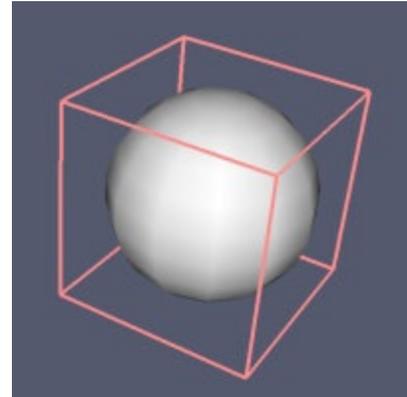
Camera Control for 3D Movements			
	Left Button	Middle Button	Right Button
	Rotate	Pan	Zoom
Shift	Roll	Rotate	Pan
Control	FlyIn	Rotate	FlyOut

You will see that there are two "styles" of mouse bindings: one for 2D movements and one for 3D movements. By default the camera is in 3D movement mode, and the Camera Control for 3D Movements area on this property sheet shows the current bindings. The default bindings are shown in the image on the right. The top row indicates what will happen when you press each of the mouse buttons in the Display Area while moving the mouse. The second row shows the motion that will occur when you press shift plus the specified mouse button, and the third row shows what will happen when pressing control plus the specified mouse button. By using the pulldown menus you can customize this interaction.

For this example, try using the left mouse button to rotate the scene. Moving the mouse in the Display Area while holding down the left mouse button will cause the camera to rotate around the scene. The point about which the camera rotates is known as the center of rotation. The middle mouse button can be used to pan (translate) the scene, and the right mouse button can be used to zoom in and out. These three operations are the most common.

### 1.4.6 Add a bounding box

In this step we are going to add a filter to the pipeline. Select the Outline item from the Filter menu on the Menu Bar. After you do this you will see the Outline property sheet appear in the Left Panel. As with sources, you need to click the Accept button to actually add this filter to the pipeline. Once you do this you will see an outline box surrounding the sphere. You will also notice that in there are now two items listed in the Selection Window: Sphere1 and Outline0. Both items are visible so an open eye is shown to the left of each. Because the Outline0 object is the active data object, it is highlighted in yellow, and if another filter is added it will be connected to this data object.



Use the Display tab for Outline0 to change the color of the line by clicking on the Actor Color button in the Color region. You can also change the thickness of the line using the Line Width thumb wheel that can be accessed by clicking on the small triangle to the right of the text entry box, or by typing a number directly in the box.

To access the properties of the Sphere1 object you can either click on Sphere1 in the Selection Window, or select this item from the Select menu on the Menu Bar. This will cause Sphere1 to be the currently active object, and the Sphere1 property sheet will be displayed. Using the Display tab change the sphere back to a surface representation. The image you have now should appear something like the one shown above.

### 1.4.7 Shrink the sphere

We will now apply another filter to the sphere. First, make sure that Sphere1 is the highlighted item in the Selection Window so that the filter we add will be connected to it. Now select the Shrink filter from the Filter menu. On the property sheet, change the Shrink Factor to 0.75 (the default is 0.5) and press Accept. You will notice that several things happen. First, a new item is added to the selection window called Shrink0 and it is now the currently selected item. In the Display Area it appears that the polygons defining the sphere have each been shrunk to 75% of their original size. Actually, the original sphere (with full size polygons) is still there, it is just not visible now. Note that the eye icon next to Sphere1 in the Selection Window is now light grey indicating that this item is invisible. When the shrink filter was added, a new data object was created based on the input data object (the sphere). This new object (the output of the shrink filter) is now visible and the original input is not visible.

Most filters do turn off the visibility of the input data object when they execute. Two exceptions to this are the outline filters (Outline and Outline Corners) since it is assumed with these filters that the user wants to see both the original data and the filter output simultaneously.

In this example you have created a branching pipeline in ParaView. You can see this in the Navigation Window. To change from Selection Window to Navigation Window, click

on the  icon next to the Selection Window title. You should see the title change to Navigation Window, and in the window you should see:



**Figure 4. Navigation window showing a pipeline with one source and one filter**

The item shown in dark grey is the currently selected data object. All other items are shown in blue indicating that you can click on that data object to select it. Click on the Sphere1 item and the display in the Navigation Window should change to this:

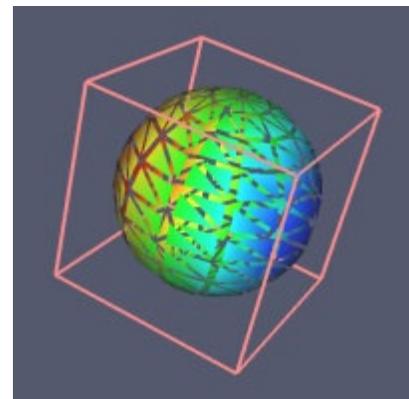


**Figure 5. Navigation window showing a branching pipeline**

This is a simple representation of the underlying VTK pipeline. We have created a `vtkSphereSource` that has output `vtkPolyData` which is used as input to both the `vtkOutlineFilter` and the `vtkShrinkFilter`.

### 1.4.8 Color by normals

Select the Shrink0 item from the Selection / Navigation Window to access the property sheet for this item. On the Display tab, change the Color By option in the Color region to Point Normals (3). Then click on the Edit Color Map... button. Select Component from the menu in the Vector portion of the user interface. Then click the Reset Range button at the top of the Color Map section. Click the Back button at the bottom of the left panel to return to the Display tab. You should now see an image similar to the one shown on the right. Instead of using a specified solid color for the color of the object, the color is derived from a lookup table based on some property of the data. In this case we have chosen to use the X value of the Normals to look up a color between blue and red.



As demonstrated, you can change the current color map using the Edit Color Map button in the View region. This will change the property sheet displayed in the Left Panel to one that can be used to control the color map and the various properties of the scalar bar. In the Color Map region you can change the two endpoint colors of the color map. An HSV interpolation strategy is employed for intermediate values.

## 1.5 Command Line Arguments

There are several command line arguments that can be used to control the execution of ParaView. These are grouped below into three categories: General, Parallel, and Mesa. When more than one option is indicated, as is the case with `--play-demo` and `-pd`, either option can be used.

### 1.5.1 General

**--start-empty, -e** : Start ParaView without any default modules. This option is useful when the user wants to customize ParaView and wants to disable all default sources, filters, readers and toolbar buttons.

**--disable-registry, -dr** : Do not use registry when running ParaView (useful for testing). With this option, ParaView will ignore all stored user settings and use the default values. No settings will be saved either.

**--play-demo, -pd** : Run the ParaView demo.

**--batch, -b** : Load and run the batch script specified.

**--stereo** : Tell the application to enable stereo rendering (single process only)

**--use-offscreen-rendering, -os** : Render offscreen on satellite processes. On Unix, this options works only with software rendering or mangled mesa.

**--render-module** : Use specified rendering module (--render-module=...)

**--help** : Displays available command line arguments.

### 1.5.2 Client/Server

**--client, -c** : Start ParaView in client mode (single process).

**--client-render-server, -crs**: Start ParaView in client mode (single process) to connect to a server and a render server.

**--server, v** : Start ParaView in server mode (single or multi-process).

**--render-server, -rs**: Start ParaView in render server mode (single or multi-process).

**--machines, -m**: Specify the network configuration file for the render server.

**--host** : Tell the client where to look for the server (default --host=localhost). Use this option with --client option or --server -rc option.

**--user** : Tell the client what username to send to server when establishing SSH connection. Only necessary to start the server remotely.

**--port** : Specify the port client and server will use (default --port=11111). Client and server ports must match.

**--render-port**: Specify the port client and render server will use (default --render-port=22221). Client and render server ports must match.

**--render-node-port**: Specify the port to be used by each render node.

**--reverse-connection, -rc** : Have the server connect to the client.

### 1.5.3 Parallel

**--use-tiled-display, -td** : Duplicate the final data to all nodes and tile node displays 1-N into one large display.

**--tile-dimensions-x, -tdx** : **-tdx=X** where **X** is number of displays in each row of the display.

**--tile-dimensions-y, -tdy** : **-tdy=Y** where **Y** is number of displays in each column of the display

### 1.5.4 Mesa

**--use-software-rendering, -r** : Use software (Mesa) rendering on all nodes. This supports offscreen rendering.

**--use-satellite-software, -s** : Use software (Mesa) rendering only on satellite processes. This is useful when the user wants to have a window only on the first node's display. This is accomplished by using this option in combination with the **--use-offscreen-rendering** option. Furthermore, since the first node uses hardware accelerated rendering, the performance is not compromised when rendering locally.

## 1.6 Running ParaView With MPI

### 1.6.1 Distributed Stand-Alone Mode

If compiled with MPI support, ParaView can be launched like any other MPI application. The method for starting an MPI application is system and implementation dependent. Contact your system administrator for information on how this can be done. Once started, the user interface will appear on the display attached to the root process (process with id 0). On the displays attached to the other processes (satellite processes), independent render windows will appear. These do not have associated user interfaces and can not be manipulated by the user. Note that all processes must have access to a display. This does not require the presence of a monitor. As long as ParaView can open windows from all processes and read their contents, it will function properly. If multiple processes share the same display, it is very likely that windows opened by these processes will overlap. If this happens, the contents of those windows can not be read by ParaView and the image on the main window (the one with the user interface) will be corrupt. The same can happen if there are other windows which hide part or all of a satellite process' window. ParaView supports off-screen rendering on satellite nodes. See the previous section for a list of command line arguments to enable this feature.

### 1.6.2 Client/Server Mode

In client/server mode, the processing of the data (on the server) is separated from the user interface (on the client). The server can be started as a single process or with multiple processes through MPI. The **--server** command line option must be used when

starting the server. The client runs in a single process and connects to the server. The --client command line option must be used when starting the client.

The server processes do not have user interfaces, and the user cannot interact with them directly. The server processes have the same display issues as mentioned in the previous section (distributed stand-alone mode).

### 1.6.3 Unix/Linux Related Issues

The environment(s) in which ALL processes run must define the DISPLAY variable. Furthermore, they must have the right access to these displays. With some MPI implementations, it is possible to pass environmental variables to MPI processes. Some MPI implementation use ssh which sets up X tunneling to allow the remote programs to create windows on the display of the system which starts the program. Unless this behavior is changed, all windows will show up on one display and most probably overlap. Our solutions to these problems are as follows.

- Create a default login on all nodes which sets the necessary variables (for example by creating/modifying .Xauthority – better solution – or using xhost +localhost – less secure solution).
- On each process, set a DISPLAY variable which depends on where the process is running and if necessary, set up authorization in a start-up file which belongs to the user which starts the MPI job (e.g., .bashrc, .cshrc, or equivalent; note that .login and .profile are not always executed when the shell is not interactive).

On Unix/Linux, ParaView supports off-screen rendering on satellite nodes via Mesa when this feature is enabled during the build process.

### 1.6.4 Windows Related Issues

In some MPI implementations, the remote tasks started by the MPI system are not allowed to interact with the desktop. Unless this is changed, ParaView can not function. The solution to this problem is implementation dependent and usually involves changing the setting of the service responsible for starting processes and sometimes recompiling the service (obviously not an option if no source code is available).

## 1.7 Check for Updates

ParaView users can stay up-to-date with new releases of ParaView by visiting the Download section of the ParaView web site.

## 1.8 ParaView Support

A searchable list of Frequently Asked Questions (FAQ) and the corresponding answers is available here: <http://www.paraview.org/cgi-bin/paraviewfaq>. This is the first place you should look if you have a question or encounter a problem.

The ParaView mailing list is the principal means of communication among users and developers. Please go to <http://www.paraview.org/mailman/listinfo/paraview> to subscribe to the list. Instructions are given to receive an archive version of the list as well as to manage and unsubscribe from the list. Many developers and users of ParaView are subscribed to the list, so it is a good resource for questions that cannot be answered using the ParaView FAQ.

Commercial support is also available for ParaView. Contact Kitware at 518-371-3971 or [kitware@kitware.com](mailto:kitware@kitware.com) for more information.

## 2 Loading Data

### 2.1 Overview

There are several ways to load data into ParaView. One method is to use the Open Data option from the File menu on the Menu Bar to specify a data file. Details on this method including the supported data types are provided in the next section.

Another method of loading information into ParaView is to load a previously saved session file by selecting the Load Session option from the File menu. Session files save all of the important state changes in ParaView into a file that can be loaded back into ParaView at a later time to re-create the state of the system. Session files are covered in more detail at the end of this chapter.

When either the Open Data or Load Session options are selected, an OS-specific dialog box will appear allowing you to select the file to load. On Windows XP, this dialog box may look similar to the example shown below. Using the tools provided by the dialog, you can filter for a specific data type, traverse the directory structure, select a file, and double-click or press the Open button. The Cancel button on the dialog can be used to cancel the load operation.

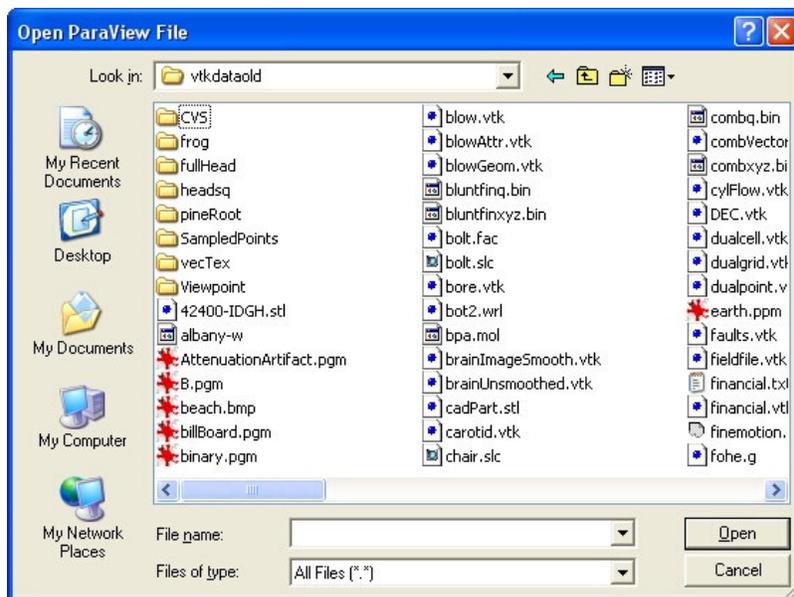


Figure 6. Open ParaView File dialog

## 2.2 Supported Data Formats

By default ParaView supports the following file types:

**ParaView Files:** This is the default file format for ParaView. The data object created by this reader may be of any type supported by ParaView (polygonal, image, rectilinear, structured grid or unstructured). The file extension is .pvd. This format supports spatially partitioned as well as multi-block data.

**VTK Files:** XML based file format used by VTK. The data object created by this reader may be of any type supported by ParaView (polygonal, image, rectilinear, structured grid or unstructured). The file extensions are: .vtp for polygonal data, .vti for image data, .vtr for rectilinear grids, .vts for structured grids, and .vtu for unstructured data.

**Parallel (partitioned) VTK Files:** XML based file format used by VTK. The parallel VTK files contain information about the spatial distribution of data and may point to multiple VTK files. The data object created by this reader may be of any type supported by ParaView (polygonal, image, rectilinear, structured grid or unstructured). The file extensions are: .pvtp for polygonal data, .pvti for image data, .pvtr for rectilinear grids, .pvts for structured grids, and .pvту for unstructured data.

**Legacy VTK Files:** This is the legacy (before VTK 4.2, although still supported) VTK file format. All types of data are stored with the same file extension. Files of this type will have a .vtk extension. The data object created by this reader may be of any type supported by ParaView (polygonal, image, rectilinear, structured grid, or unstructured).

**Parallel (partitioned) legacy VTK Files:** This is the legacy (before VTK 4.2, although still supported) VTK file format. All types of data are stored with the same file extension. Files of this type will have a .pvtk extension. The data object created by this reader may be of any type supported by ParaView (polygonal, image, rectilinear, structured grid, or unstructured).

**EnSight Files:** This is the file format used by CEI's EnSight (<http://www.ensight.com>). ASCII and binary EnSight 6 and EnSight Gold formats are supported. These files will have a .case extension. The data object created by this reader may be of any type supported by ParaView (polygonal, image, rectilinear, structured grid, or unstructured). This format also supports multiple parts.

**EnSight Master Server Files:** This is the file format used by CEI's EnSight (<http://www.ensight.com>) for spatially distributed (partitioned) data. The master file usually has a .sos extension and points to multiple EnSight case files.

**HDF5 Files:** Files of this type will have a .h5 extension. This format only supports image data.

**VRML Files:** This is the file format for the Virtual Reality Modeling Language. VRML 2.0 format is supported. Only the geometry from the VRML file is loaded.

**PLOT3D Files:** This is the file format originally used by the PLOT3D plotting package developed at NASA. The default file extensions ParaView assumes are .xyz for the geometry file and .q for the solution file. Only structured grid (binary, ASCII, single, or multi-block) data format is supported.

**Stereo Lithography:** ParaView can read binary or ASCII Stereo Lithography files. These files will have a .stl extension. This format supports only polygonal data.

**BYU Files:** ParaView can read MOVIE.BYU files. These files will have a .g extension. This format supports only polygonal data.

**Protein Data Bank Files:** File format used by Protein Data Bank (PDB), an archive of experimentally determined three-dimensional structures of biological macromolecules (<http://www.rcsb.org/pdb/>). These files will have a .pdb extension. The PDB reader produces polygonal data output.

**XMol Files:** Minnesota Supercomputer Center's XMol file format. XYZ is a simple file format for representing molecules. It describes atoms and bonds, but no values are stored on atoms. These files will have a .xyz extension. The XMol reader produces polygonal data output.

**XDMF Files:** The eXtensible Data Model and Format (XDMF) is an active, common data hub used to pass values and metadata in a standard fashion between application modules (<http://www.arl.hpc.mil/ice/>). These files will have an .xmf extension. This format supports rectilinear and unstructured grids.

**Raw (Binary) Files:** ParaView supports reading raw regular rectilinear grid data from a file.

**Exodus Files:** ParaView can read ExodusII files. The expected file extension is .ex2.

**SAF Files:** ParaView can read SAF files. The SAF reader produces image, rectilinear grid, and unstructured grid output.

**AVS UCD Files:** ParaView can read binary or ASCII files stored in AVS UCD format. The output of the AVS UCD reader is of type unstructured grid.

**Meta Image Data Files:** ParaView can read UNC meta image data. This reader produces image data output.

**Gaussian Cube Files:** This is the file format used by the Gaussian software package (<http://www.gaussian.com>). The default file extension is .cube. The output produced is polygonal.

Support for other file formats can be added by either writing a simple XML configuration file (if the VTK reader already exists) or by first implementing a VTK reader in C++ and then adding it to the XML file for ParaView readers.

## 2.3 Data Types

The data types supported by ParaView are

**Image Data** (uniform rectilinear grid or volume): An image dataset is a collection of points and cells arranged on a regular, rectangular lattice. The rows, columns and planes of the lattice are parallel to the global x-y-z coordinate system.

**Rectilinear Grid** (non-uniform rectilinear grid): The rectilinear dataset is a collection of points and cells arranged on a regular lattice. The rows, columns and planes of the

lattice are parallel to the global x-y-z coordinate system. While the topology of the dataset is regular, the geometry is only partially regular. That is, the points are aligned along the coordinate axes, but spacing between points may vary.

**Structured Grid** (curvilinear grid): A structured grid is a dataset with regular topology and irregular geometry. The grid may be warped into any configuration in which the cells do not overlap or intersect. The topology of the structured grid is represented implicitly by specifying a 3-vector of dimensions ( $n_x, n_y, n_z$ ). The geometry is explicitly represented by maintaining an array of point coordinates.

**Unstructured Grid:** The most general form of dataset is the unstructured grid. Both the topology and geometry are completely unstructured. Cells of any type can be combined in arbitrary combinations in an unstructured grid.

**Polygonal Data:** Polygonal data is a collection of geometric primitives usually used by the rendering engine to generate the images displayed in the Display Area. The polygonal dataset consists of vertices, polyvertices, lines, polylines, polygons and triangle strips.

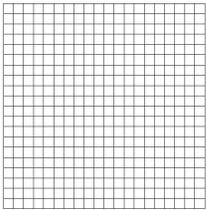
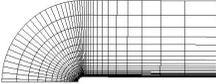
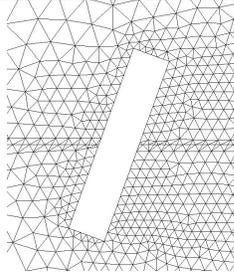
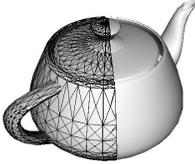
		
Image Data	Rectilinear Grid	Structured Grid
		
Unstructured Grid	Polygonal Data	

Figure 7. ParaView data types

## 2.4 ParaView Session Files

Using the Load Session option from the File menu will cause the Load Script dialog box to pop up. Using this dialog you can select a ParaView Session file to load. These session files will have a .pvs extension.

There are two types of session files:

**Session State:** These files contain commands to recreate the state of ParaView at the time of the creation of the session file. They do not contain information about the steps that did not contribute to the final state.

**Session Trace:** These files contain the sequence of all steps that were performed in ParaView to reach the current state.

## 3 Sources

### 3.1 Overview

The Source menu in the ParaView Menu Bar contains a list of sources that can be added. When you select an item from the list, several things will occur:

- The property sheet for the source will be displayed. On this property sheet you will see the configurable parameters of the source.
- The Selection / Navigation Window will be displayed if it was not displayed already.
- The source that you are creating will be the currently active data object.

The source will not be created and displayed until you press the Accept button, which is highlighted in green whenever a source is first created or a parameter is changed that would require updating the source. If you would like to cancel the creation of the source, press the Delete button. To reset all the parameters back to the default values (during creation) or to the values they had when the Accept button was last pressed (during modification) press the Reset button.

The property sheet that is displayed when a Sphere source is created is shown on the right. The name Sphere1 is a unique identifier for this object. The class indicates the type of VTK object. The name of the object in the Selection / Navigation Window can be changed by typing a new label in the text entry box. A brief description of the source is given below the label. Below the Accept, Reset, and Delete buttons are the parameters specific to this type of source.

The screenshot shows the property sheet for a Sphere source. At the top, it displays the Name as 'Sphere1', the Class as 'vtkSphereSource', and a text box for the Label containing 'Sphere1'. Below this is a description: '3D sphere given a center and radius.' There are three buttons: 'Accept' (highlighted in green), 'Reset', and 'Delete'. The parameter section includes:

Center	0	0	0
Radius	0.5		
Theta Resolution			8
Start Theta			0
End Theta			360
Phi Resolution			8
Start Phi			0
End Phi			180

The Accept button is used to create the source and to accept any future parameter changes that you make. Changing a value such as the radius of the sphere will have no effect until the Accept button is pressed. Note that the Accept button will turn green to indicate that something has changed requiring the source to update. The Reset button

can be used to reset all parameters back to the values used during the last update (or the default values if the source has not been created). The Delete button can be used to delete this source only if it is not used as input to a filter. If the Delete button is disabled, it indicates that a filter relies on the output of this source and therefore this source cannot be deleted until after the filter using its output is deleted.

The remaining sections in this chapter of the ParaView User's Guide will cover several of the source objects available in ParaView.

## 3.2 3D Text

The 3D Text source can be used to add polygonal text into the 3D scene. By default, the text will have an origin of (0,0,0) and will lie on the  $z=0$  plane. The text can be translated, rotated, and scaled using the Actor control area of the Display tab for this source, and it can be interactively positioned using the Move operation of the camera. The object created by this source is a **vtkVectorText** object, and the output of this source is **vtkPolyData**. An example of 3D Text is given below.

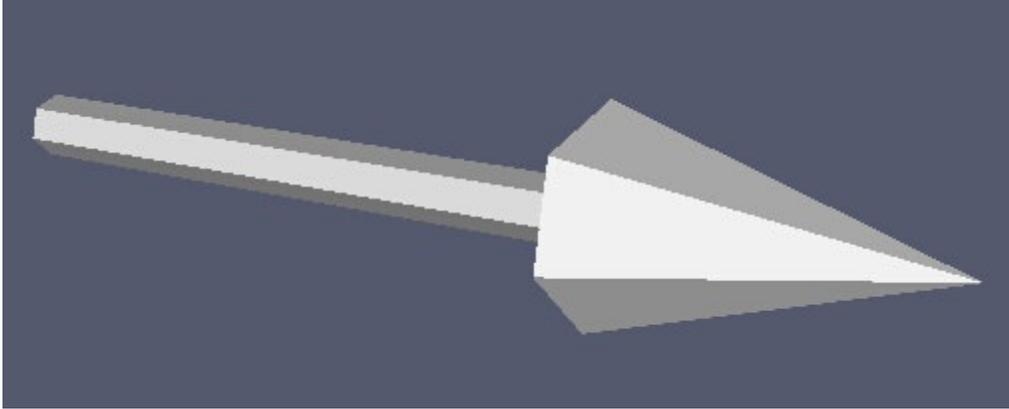


The 3D Text source has only one parameter:

**Text:** Enter the text string in this entry box. Besides the ASCII alphanumeric characters a-z, A-Z, 0-9, this source also supports ASCII punctuation marks.

## 3.3 Arrow

The Arrow source can be used to add a polygonal arrow to the 3D scene. The arrow consists of a cylinder for the shaft, and a cone for the tip. By default the arrow has an origin at (0,0,0), has a length of 1.0, and is pointing along the positive x axis. The arrow can be translated, rotated, and scaled using the Actor control area of the Display tab for this source, and it can be interactively positioned using the Move operation of the camera. The object created by this source is a **vtkArrowSource** object, and the output of this source is **vtkPolyData**. An example of an Arrow source is given below.



The Arrow source has the following parameters:

**Tip resolution:** This is the number of faces around the cone representing the tip of the arrow. A larger number will mean a smoother cylinder but more polygons which can lead to slower rendering times. The default value is 6 which is fairly coarse.

**Tip radius:** This is the radius of the cone defining the tip of the arrow. The length of the whole arrow is 1.0 unit. By default the tip radius is .1 unit.

**Tip length:** This is the length of the tip of the arrow and should be a number greater than 0.0 and less than 1.0. The total length of the arrow is 1.0 unit, so if the tip length is set to 0.25, then the shaft cylinder will be 0.75 units long. The default value is 0.35.

**Shaft resolution:** This is the number of faces around the cylinder representing the shaft of the arrow. The default value is 6.

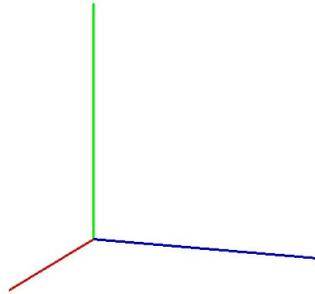
**Shaft radius:** This is the radius of the cylinder defining the shaft of the arrow. The default value is 0.03.

The output polydata from the Arrow source will not contain normals, therefore rendering of the arrow will be performed using flat shading. The appearance of the arrow can be improved without significantly increasing the resolution of the tip and shaft by generating normals (Filter menu | Normals generation).

## 3.4 Axes

The Axes source can be used to add a representation of the coordinate system axes to the 3D scene. The X axis will be drawn as a blue line, the Y axis as a green line, and the Z axis as a red line. The axes can be drawn as three lines drawn in the positive direction from the origin, or three lines crossing at the origin (drawn in both the positive and negative directions). The origin and scale of the axes can be changed through the parameters. The axes can be translated, rotated, and scaled using the Actor control area of the Display tab for this source, and it can be interactively positioned using the Move operation of the camera. The object created by this source is a **vtkAxes** object, and the output of this source is **vtkPolyData**. The output polydata has a scalar per line

so that the lines can be colored. The output polydata also has normals defined. An example of an axes source is given below.



The Axes source has the following parameters:

**Scale:** By default the axes lines have a length of 1, or if the symmetric option is selected, a length of 1 in each direction for a total length of 2. This value can be changed to make the axes bigger or smaller.

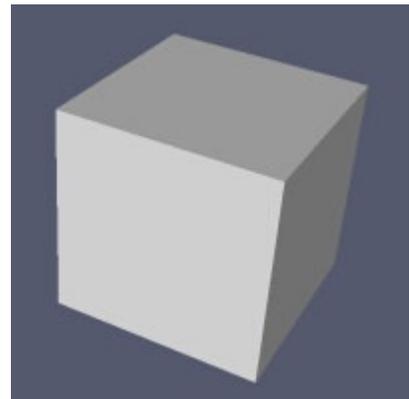
**Origin:** These entry boxes can be used to change the origin of the axes. The origin of the axes is at (0,0,0) by default.

**Symmetric:** When the symmetric option is selected, the axes extend along each of the positive and negative directions for a distance equal to the scale value. When unchecked (the default) the axes extend only in the positive direction.

Note that by default the axes will appear to change color as you rotate. To display the axes in solid color lines, use the Display Style area on the Display tab and change the representation to wireframe. In the example image shown above, the representation was changed to wireframe and the background color was changed to white to improve the clarity of the image.

## 3.5 Box

The Box source can be used to add a box to the 3D scene. The center of the box and the X, Y, and Z lengths of the box can be changed using the property sheet. The box can be translated, rotated, and scaled using the Actor control area of the Display tab for this source, and it can be interactively positioned using the Move operation of the camera. The object created by this source is a **vtkCubeSource** object, and the output of this source is **vtkPolyData**. The output polydata has both normals and texture coordinates defined. An example of a box source is given on the right.



The Box source has the following parameters:

**X Length:** This value specifies the length of the box along the X axis. By default the X length of the box is 1.

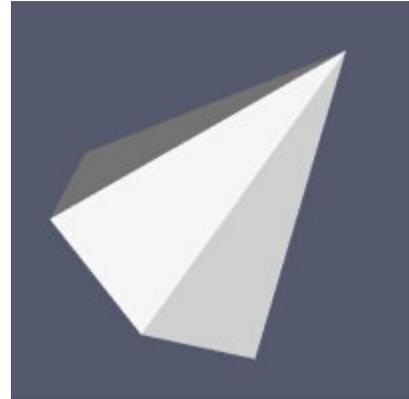
**Y Length:** This value specifies the length of the box along the Y axis. By default the Y length of the box is 1.

**Z Length:** This value specifies the length of the box along the Z axis. By default the Z length of the box is 1.

**Center:** The three values represent the coordinate at the center of the box. By default the box is centered at (0,0,0).

## 3.6 Cone

The Cone source can be used to add a polygonal cone to the 3D scene. The radius and height of the cone, the resolution of the polygonal approximation, and whether or not the cone is capped can be changed using the property sheet. The cone can be translated, rotated, and scaled using the Actor control area of the Display tab for this source, and it can be interactively positioned using the Move operation of the camera. The object created by this source is a **vtkConeSource** object, and the output of this source is **vtkPolyData**. An example of a cone source is given on the right.



The Cone source has the following parameters:

**Resolution:** This value indicates the number of divisions around the cone. The higher this number, the closer the polygonal approximate will come to representing a cone, and the more polygons it will contain. The default resolution is 6.

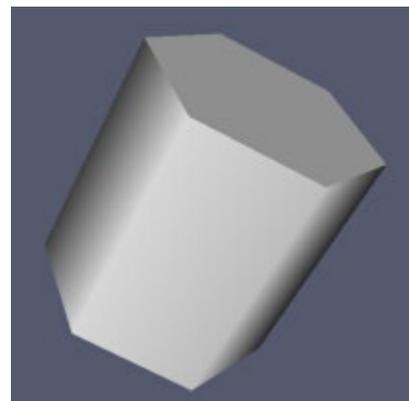
**Radius:** This is the radius of the cone. The default value is 0.5.

**Height:** This is the height of the cone. The default value is 1.0.

**Capping:** This check box indicates whether the cone is capped (the default) or open. If the cone is capped, there is an N sided polygon closing off the cone where N is the resolution.

## 3.7 Cylinder

The Cylinder source can be used to add a polygonal cylinder to the 3D scene. The radius, height, and center of the cylinder, the resolution of the polygonal approximation, and whether or not the cylinder is capped can be changed using the property sheet. The cylinder can be translated, rotated, and scaled using the Actor control area of the Display tab for this source, and it can be interactively positioned using the Move operation of the camera. The object created by this source is a **vtkCylinderSource** object, and the output of this source



is **vtkPolyData**. The output polydata has normals and texture coordinates defined. An example of a cylinder source is given on the right.

The Cylinder source has the following parameters:

**Resolution:** This value indicates the number of divisions around the cylinder. The higher this number, the closer the polygonal approximate will come to representing a cylinder, and the more polygons it will contain. The default resolution is 6.

**Radius:** This is the radius of the cylinder. The default value is 0.5.

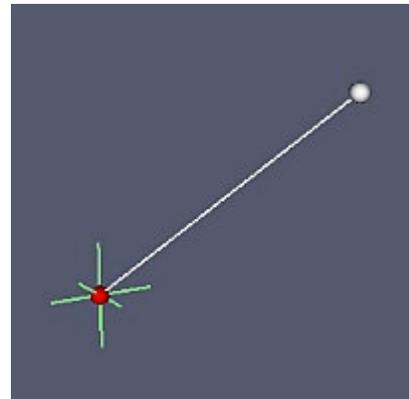
**Height:** This is the height of the cylinder. The default value is 1.0.

**Center:** These three values represent the coordinate value at the center of the cylinder. By default, the cylinder is centered on (0,0,0).

**Capping:** This check box indicates whether the cylinder is capped (the default) or open. If the cylinder is capped, there are two N sided polygons closing off the cylinder where N is the resolution.

## 3.8 Line

The Line source can be used to interactively or manually add a line to the 3D scene. The two end points of the line, the resolution of the line, and the visibility of the interaction widget can be changed using the property sheet. In addition, the two end points can be specified using the interaction widget. The line can be translated, rotated, and scaled using the Actor control area of the Display tab for this source, and it can be interactively positioned using the Move operation of the camera. The object created by this source is a **vtkLineSource** object, and the output of this source is **vtkPolyData**. An example of a line source is given on the right. Note that in this example the interaction widget is visible, and the lower left point of the line is being manipulated. Widget interaction is described below.



The Line source has only one parameter:

**Resolution:** This value represents the number of line segments in line. By default this value is 1. As opposed to some other sources such as the sphere, cone, and cylinder, increasing the resolution does not increase the accuracy of the line, it simply increases the number of colinear line segments defining the line. This can sometimes be useful when using the data as input to a filter.

The Line source interaction widget has the following parameters which are displayed on the Line source property sheet:

**Point1:** These three values indicate the coordinates of one of the two end points of the line.

**Point2:** These three values indicate the other end point coordinate of the line.

**Visibility:** This check box can be used to toggle the visibility of the interaction widget. When the interaction widget is visible, you will see a thick line with two spheres at the end points. To see the actual line you must turn off the visibility of the interaction widget. By default the interaction widget is visible. This check box is tied to the interaction widget and not the source object, and therefore has an immediate effect; the Accept button does not need to be pressed.

The Line source interaction widget has the behavior described below. Changes made through the interaction widget must be accepted by pressing the Accept button on the property sheet before the change will be reflected in the output of the line source.

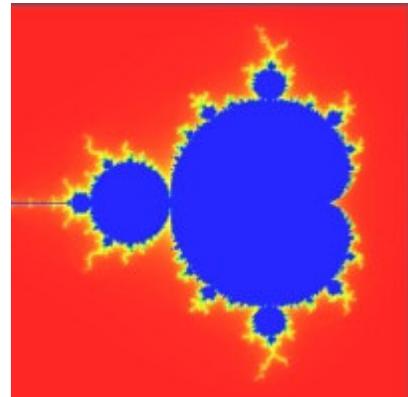
**Left Mouse on End Point Sphere:** Clicking the left mouse button while the cursor is over one of the end point spheres, then dragging the mouse will cause that end point to translate.

**Left Mouse on Line:** Clicking the left mouse button while the cursor is over the line, then dragging the mouse will cause the line to translate.

**Right Mouse on Line:** Clicking the right mouse button while the cursor is over the line, then dragging the mouse will cause the line to increase or decrease in length. Upward mouse motion will cause an increase in length; downward motion will cause a decrease in length.

## 3.9 Mandelbrot

The Mandelbrot source can be used to add a regular rectilinear grid with scalar values derived from the Mandelbrot set to the 3D scene. The values in the grid are the number of iterations it takes for the magnitude of the value to get over 2. The equation repeated is  $z = z^2 + C$  ( $z$  and  $C$  are complex). Initial value of  $z$  is zero, the real value of  $C$  is mapped onto the  $x$  axis, and the imaginary value of  $C$  is mapped onto the  $Y$  Axis. The third dimension ( $z$  axis) is the imaginary value of the initial value. By default a two dimensional data set is created with 251x251 values on the  $z=0$  plane. The size of the output data as well as the parameters for generating the data can be controlled using the property sheet for the Mandelbrot source. The output Mandelbrot image can be translated, rotated, and scaled using the Actor control area of the Display tab for this source, and it can be interactively positioned using the Move operation of the camera. The object created by this source is a **vtkImageMandelbrotSource** object, and the output of this source is **vtkImageData**. An example of a Mandelbrot source is given on the right.



The Mandelbrot source has the following parameters:

**Extent:** These six values indicate the  $X$ ,  $Y$ , and  $Z$  extent of the output data. The first two numbers are the minimum and maximum  $X$  extent, the next two are the minimum and maximum  $Y$  extent and the final two are the minimum and maximum

Z extent. The numbers are inclusive, so the default values of 0, 250, 0, 250, 0, 0 indicate that a single image of 251x251 values will be created.

**Sub-space:** These three values allow you to set the projection from the 4D space to the axes of the 3D Volume. By default, the real component of C (represented by 0) is mapped to the X axis, the imaginary component of C (represented by 1) is mapped to the Y axis, and the real component of X, the initial value (represented by 2) is mapped to the Z axis. The imaginary component of X is represented by 3 and by default is not mapped. All values entered must be between 0 and 3 inclusive.

**Origin:** The four values indicate the values of C (real and imaginary) and the initial value X (real and imaginary). The first two numbers represent the real and imaginary components of C, and the last two indicate the real and imaginary components of X.

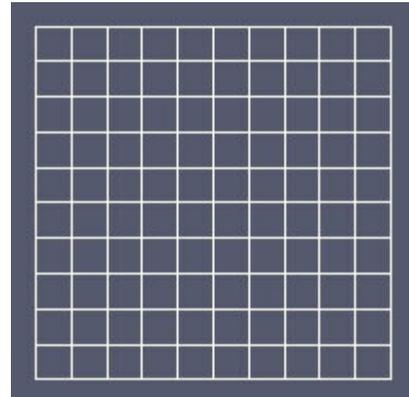
**Size:** These four values indicate the four dimensional size of which to create the output image.

**Max Iterations:** This number indicates the maximum number of iterations to perform in order to determine if the value will go above 2.

## 3.10 Plane

The Plane source can be used to add a polygonal parallelogram to the 3D scene. The origin and two other corner points that define the parallelogram, as well as the resolution of the approximation can be specified using the property sheet. As opposed to the sphere, cone, or cylinder sources, the parallelogram is exactly represented at the lowest resolution. Higher resolutions may be desired if this plane is to be used as input for a filter. The plane can be translated, rotated, and scaled using the Actor control area of the Display tab for this source, and it can be interactively positioned using the Move operation of the camera. The object created by this source is a

**vtkPlaneSource** object, and the output of this source is **vtkPolyData**. . An example of a plane source (drawn in wireframe) is given on the right. In this example, the resolution was set to 10 in X and 10 in Y leading to the 100 subdivisions that you see in the example image.



The Plane source has the following parameters:

**Origin:** The origin is the location of one of the corners of the parallelogram.

**First Point:** This coordinate represents the location of a second corner of the parallelogram. One edge of the parallelogram will be the line connecting the origin with this first point, and this will be considered the X axis of the parallelogram.

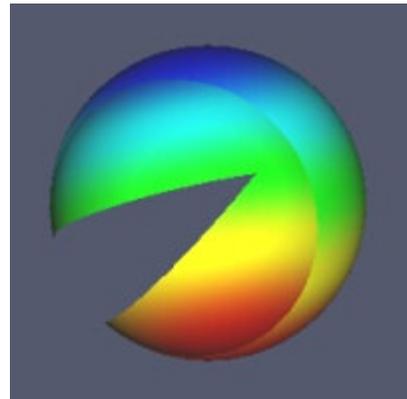
**Second Point:** This coordinate represents the location of a third corner of the parallelogram. One edge of the parallelogram will be the line connecting the origin with this second point, and this will be considered the Y axis of the parallelogram.

**X Resolution:** This is the number of divisions along the X axis of the parallelogram. By default this value is 1.

**Y Resolution:** This is the number of divisions along the Y axis of the parallelogram. By default this value is 1.

## 3.11 Sphere

The Sphere source can be used to add a polygonal sphere to the 3D scene. The center and radius of the sphere, the resolution of the polygonal approximation, and the starting / ending angles can be changed using the property sheet. The sphere can be translated, rotated, and scaled using the Actor control area of the Display tab for this source, and it can be interactively positioned using the Move operation of the camera. The object created by this source is a **vtkSphereSource** object, and the output of this source is **vtkPolyData**. The output polydata has point normals defined. An example of a sphere source is given on the right. For this example, the resolutions were changed to 200, the End Theta angle was set to 320, the End Phi angle was set to 120, and the sphere is colored by the Y-component of the Point Normals.



The Sphere source has the following parameters:

**Center:** This coordinate represents the center of the sphere. By default this value is (0,0,0).

**Radius:** This is the radius of the sphere. The default value is 0.5.

**Theta Resolution:** This number represents the number of division between start theta and end theta around the sphere. The theta divisions are similar to longitude lines on the earth. The higher the resolution the closer the approximation will come to a sphere and the more polygons there will be. The default theta resolution is 8.

**Start Theta:** To form a complete sphere the start theta should be 0 and the end theta should be 360. The start theta can be adjusted to form only a portion of a sphere. The default value is 0.

**End Theta:** The end theta can be adjusted to form only a portion of a sphere as shown in the example above. The default end theta value is 360.

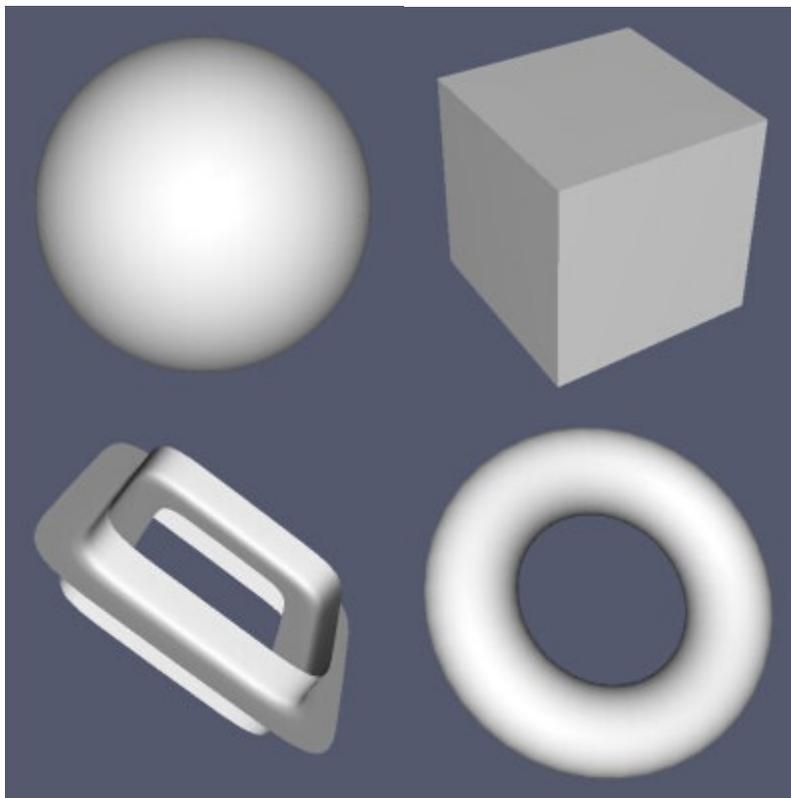
**Phi Resolution:** This number represents the number of division between start phi and end phi on the sphere. The phi divisions are similar to latitude lines on the earth. The default phi resolution is 8.

**Start Phi:** To form a complete sphere the start phi should be 0 and the end phi should be 180. The start phi can be adjusted to form only a portion of a sphere. The default value is 0.

**End Phi:** The end phi value can be adjusted to form only a portion of a sphere. In the example above both the end theta end phi values were reduced to leave a gap in the sphere both in the longitudinal and latitudinal directions. The default end phi value is 180.

## 3.12 Superquadric

The Superquadric source can be used to add a polygonal superquadric to the 3D scene. The resolution in both the latitude (phi) and longitude (theta) directions can be specified. Roundness parameters control the shape of the superquadric. The Toroidal boolean controls whether a toroidal superquadric is produced. If so, the Thickness parameter controls the thickness of the toroid: 0 is the thinnest allowable toroid, and 1 has a minimum sized hole. The resulting superquadric can be translated, rotated, and scaled using the Actor control area of the Display tab for this source, and it can be interactively positioned using the Move operation of the camera. The object created by this source is a **vtkSuperquadricSource** object, and the output of this source is **vtkPolyData**. The output polydata has point normals and texture coordinates defined. Four example superquadric objects are given below. Note that this is a powerful source object in that it can be used to create a sphere, a box, a torus, or a variety of other shapes simply by adjusting the roundness parameters.



The Superquadric source has the following parameters:

**Center:** This coordinate represents the center of the superquadric. By default this value is (0,0,0).

**Scale:** These three values can be used to scale the superquadric in X, Y, and Z. The normals will be computed correctly even with anisotropic scaling. By default all three values are set to 1.

**Theta Resolution:** This number represents the number of division in the theta (longitudinal) direction. The default theta resolution is 16. This value will be rounded to the nearest multiple of 8.

**Phi Resolution:** This number represents the number of division in the phi (latitudinal) direction. The default phi resolution is 16. This number will be rounded to the nearest multiple of 4.

**Thickness:** If the Toroidal box is checked, this value represents the thickness of the object as a value between 0 and 1. A value close to 0 leads to a thin object with a large hole, and a value near 1 leads to a thick object with a very small hole.

**Theta Roundness:** Define the roundness in the theta direction. A value of 0 represents a rectangular shape, a value of 1 represents a circular shape, and values greater than 1 produce higher order shapes. The default value is 1.

**Phi Roundness:** Define the roundness in the phi direction. A value of 0 represents a rectangular shape, a value of 1 represents a circular shape, and values greater than 1 produce higher order shapes. The default value is 1.

**Size:** This value represents an isotropic size of the superquadric. By default it is 0.5. Note that the Size and Thickness parameters control coefficients of superquadric generation, and may not exactly describe the size of the superquadric.

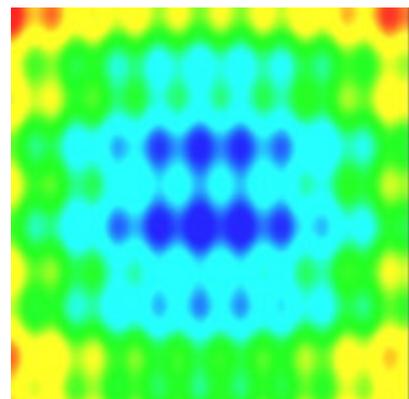
**Toroidal:** If this box is not checked (the default) the generated superquadric will not contain a hole. If checked, a toroidal object is generated.

### 3.13 Wavelet

The Wavelet source can be used to create a regular rectilinear grid in up to three dimensions with values varying according to a periodic function. The function that is evaluated is:

$$M * G * (XM*\sin(XF*x) + YM*\sin(YF*y) + ZM*\cos(ZF*z))$$

where M represents the Maximum value, G represent the Gaussian, XM, YM, and ZM are the X, Y, and Z magnitude values and XF, YF, and ZF are the X, Y, and Z frequency values. If a two dimensional extent is specified (as in the example on the right) the resulting image will be



displayed. If a three dimensional extent is used, then the bounding box of the volume will be displayed. The image or volume can be translated, rotated, and scaled using the Actor control area of the Display tab for this source, and it can be interactively positioned using the Move operation of the camera. The object created by this source is a **vtkRTAnalyticSource** object, and the output of this source is **vtkImageData**.

The Wavelet source has the following parameters:

**Extent:** These six values indicate the X, Y, and Z extent of the output data. The first two values represent the minimum and maximum X indices, the next two are the minimum and maximum Y indices, the last two are the minimum and maximum Z indices. By default each axis ranges from -10 to 10.

**Center:** This coordinate represents the center of the output data. By default this is (0,0,0).

**Maximum:** This is a scale on the output. The default is 255.

**X Freq.:** This is the XF value from the equation above. The default value is 60.

**Y Freq.:** This is the YF value from the equation above. The default value is 30.

**Z Freq.:** This is the ZF value from the equation above. The default value is 40.

**X Mag.:** This is the XM value from the equation above. The default value is 10.

**Y Mag.:** This is the YM value from the equation above. The default value is 18.

**Z Mag.:** This is the ZM value from the equation above. The default value is 5.

**Standard Dev.:** The standard deviation is used in the generation of the Gaussian value. The default value is 0.5.

## 4 Filters

### 4.1 Overview

The Filter menu in the ParaView Menu Bar contains a list of filters that can be applied to the active data object. In addition, a subset of these filters is available from the toolbar. The list of filters available in the Filter menu and the set of filters that are active on the Toolbar will change depending on the type of the currently active data object. When you select an item from the Filter menu or press a filter button in the Toolbar, several things will occur:

- The property sheet for the filter will be displayed. On this property sheet you will see the configurable parameters of the filter.
- The Selection / Navigation Window will be displayed if it was not yet displayed already.
- The filter that you are creating will be the currently active data object.

The filter will not be created and displayed until you press the Accept button, which is highlighted in green whenever a filter is first created or a parameter is changed that

would require updating. If you would like to cancel the creation of the filter, press the Delete button. To reset all the parameters back to the default values (during creation) or to the values they had when the Accept button was last pressed (during modification) press the Reset button.

A brief description of several of the filters in ParaView follows.

## 4.2 Calculator

The Calculator filter can be used to create new arrays by performing mathematical operations on existing arrays. Using an interface similar to a calculator you can specify an equation to be applied to specified data array(s) of the input data object. The Calculator is available on the Toolbar.

## 4.3 Cell Centers

The Cell Centers filter takes as input any data set and generates an output point at the center of each cell in the data set. These points can be used for placing glyphs or labeling. The center is the parametric center of the cell, not necessarily the geometric or bounding box center. The cell attributes of the input data set will be associated with the output points.

## 4.4 Cell Data to Point Data

The Cell Data to Point Data filter transforms cell data (i.e., data specified per cell) into point data (i.e., data specified at cell points). The method of transformation is based on averaging the data values of all cells that use a particular point. Optionally, the input cell data can be passed through to the output as well.

## 4.5 Clean

The Clean filter takes polygonal data as input and generates polygonal data as output. `vtkCleanPolyData` can merge duplicate points within a specified tolerance, eliminate points that are not used, and transform degenerate cells into appropriate forms. (For example, a triangle is converted into a line if two points of the triangle are merged.)

## 4.6 Clip

The Clip filter will clip a data set using a plane, a sphere, a box, or scalars. This will not decrease the dimensionality of the data object being clipped. For example, clipping a sphere with a plane may produce a hemisphere. The Clip filter is available on the Toolbar.

## 4.7 Connectivity

The Connectivity filter extracts cells that share common points and/or meet other connectivity criteria. (Cells that share vertices and meet other connectivity criteria such as scalar range are known as a region.) The filter extracts the largest connected region in the data set.

## 4.8 Contour

The Contour filter can be used to generate isosurfaces (from three dimensional data) or isolines (from two dimensional data). A set of scalar values can be specified to extract more than one contour at a time. New normals, gradients, and scalars can be optionally generated. The Contour filter is available on the Toolbar.

## 4.9 Crop

This filter will extract a volume / area of interest out of a larger structured data set.

## 4.10 Cut

The Cut filter can be used to cut a data object with a plane or a sphere. This will usually decrease the dimensionality of the data object being cut. For example, cutting a sphere with a plane may produce a circle. An interactive widget can be used for placing the plane or sphere, and a set of offset distances can be specified to perform more than one cut at a time. The Cut filter is available on the Toolbar.

## 4.11 Decimate

The Decimate filter can be used to reduce the number of triangles in a triangle mesh, while still maintaining a good approximation to the original geometry. The input to `vtkDecimatePro` is a `vtkPolyData` object, and only triangles are treated. If you desire to decimate polygonal meshes, first triangulate the polygons with `vtkTriangleFilter` object (using the Triangulate entry in the Filter menu).

The implementation of `vtkDecimatePro` is similar to the algorithm originally described in "Decimation of Triangle Meshes", Proceedings of Siggraph '92, except that this algorithm does not necessarily preserve the topology of the mesh, and it is guaranteed to give the mesh reduction factor specified by the user.

The algorithm proceeds as follows. Each vertex in the mesh is classified and inserted into a priority queue. The priority is based on the error to delete the vertex and retriangulate the hole. Vertices that cannot be deleted or triangulated (at this point in the algorithm) are skipped. Then each vertex in the priority queue is processed (i.e., vertex deletion followed by hole triangulation using edge collapse). This continues until the priority queue is empty. Next, all remaining vertices are processed. The mesh is split

into separate pieces along sharp edges or at non-manifold attachment points, and the vertices are reinserted into the priority queue. Again, the priority queue is processed until empty. If the desired reduction is still not achieved, the remaining vertices are split as necessary (in a recursive fashion) so that it is possible to eliminate every triangle as necessary.

To use this filter you should specify the Target Reduction. The algorithm is guaranteed to generate a reduced mesh at this level as long as the following four conditions are met: 1) topology modification is allowed (Preserve Topology off); 2) mesh splitting is enabled (on by default; not modifiable from within ParaView); 3) the algorithm is allowed to modify the boundary of the mesh (Boundary deletable on); and 4) the maximum allowable error is set to VTK\_DOUBLE\_MAX (the default; not modifiable from within ParaView). The Feature angle parameter can impact the quality of the final mesh.

## **4.12 Elevation**

The Elevation filter can be used to generate scalar values for a data set. The scalar values lie within a user-specified range, and are generated by computing a projection of each data set point onto a line. The line can be oriented arbitrarily. A typical example is to generate scalars based on elevation or height above a plane.

## **4.13 Extract Edges**

The Extract Edges filter will extract the edges of a 2D or 3D data set.

## **4.14 Extract Surface**

The Extract Surface filter will extract a 2 dimensional boundary surface of a data set using neighborhood relations to eliminate internal surfaces.

## **4.15 Extract Grid**

The Extract Grid filter allows you to extract a sub-grid from an input structured data object. The minimum and maximum extents of the volume of interest in X, Y, and Z can be specified along with a sampling rate. Sampling rates greater than one will subsample the original data. The Extract Grid filter can be accessed through the Toolbar or the Filter menu.

## **4.16 Feature Edges**

The Feature Edges filter can be used to extract special types of edges from input polygonal data. These edges are either 1) boundary (used by one polygon) or a line cell; 2) non-manifold (used by three or more polygons); 3) feature edges (edges used by two triangles and whose dihedral angle  $>$  FeatureAngle); or 4) manifold edges (edges

used by exactly two polygons). These edges may be extracted in any combination. Edges may also be "colored" (i.e., scalar values assigned) based on edge type. The cell coloring is assigned to the cell data of the extracted edges

## 4.17 Normals generation

The Normals generation filter computes point normals for a polygonal mesh. The filter can reorder polygons to insure consistent orientation across polygon neighbors. Sharp edges can be split and points duplicated with separate normals to give crisp (rendered) surface definition. It is also possible to globally flip the normal orientation.

The algorithm works by determining normals for each polygon and then averaging them at shared points. When sharp edges are present, the edges are split and new points generated to prevent blurry edges (due to Gouraud shading).

Normals are computed only for polygons and triangle strips. Normals are not computed for lines or vertices.

Triangle strips are broken up into triangle polygons. You may want to restrip the triangles.

## 4.18 Glyph

The Glyph filter will create a glyph at each point in the input data object. The glyph can be a cone, a sphere, a line, an arrow, or a 2D glyph. The glyph can be oriented according to a vector in the input data object and scaled according to scalar values, vector magnitudes or vector components. An upper bound can be set on the number of glyphs to be created. The Glyph filter can be accessed through the Toolbar.

## 4.19 Linear Extrusion

The Linear Extrusion filter is a modeling filter that takes polygonal data as input and generates polygonal data on output. The input data set is swept according to some extrusion function and creates new polygonal primitives. These primitives form a "skirt" or swept surface. For example, sweeping a line results in a quadrilateral, and sweeping a triangle creates a "wedge".

There are a number of control parameters for this filter. You can control whether the sweep of a 2D object (i.e., polygon or triangle strip) is capped with the generating geometry. Also, you can specify a vector for the direction of extrusion. The amount of extrusion is controlled by the scale factor.

The skirt is generated by locating certain topological features. Free edges (edges of polygons or triangle strips only used by one polygon or triangle strips) generate surfaces. This is true also of lines or polylines. Vertices generate lines.

This filter can be used to create 3D fonts and 3D irregular bar charts or to model 2 1/2D objects like punched plates. It also can be used to create solid objects from 2D polygonal meshes.

Some polygonal objects have no free edges (e.g., sphere). When swept, this will result in two separate surfaces if capping is on, or no surface if capping is off.

## 4.20 Loop Subdivision

The Loop Subdivision filter is an approximating subdivision scheme that creates four new triangles for each triangle in the mesh. The user can specify the number of subdivisions. Loop's subdivision scheme is described in: Loop, C., "Smooth Subdivision surfaces based on triangles," Masters Thesis, University of Utah, August 1987. For a nice summary of the technique see Hoppe, H., et al, "Piecewise Smooth Surface Reconstruction," *Proceedings of Siggraph 94 (Orlando, Florida, July 24-29, 1994)*. In *Computer Graphics Proceedings, Annual Conference Series, 1994, ACM SIGGRAPH*, pp. 295-302.

The filter only operates on triangles. The point data for the output of this filter is interpolated from the input point data. New triangles created at a subdivision step will have the cell data of their parent cell.

## 4.21 Mask Points

The Mask Points filter is a filter that passes through points and point attributes from an input data set. (Other geometry is not passed through.) It is possible to mask every nth point, and to specify an initial offset to begin masking from. A special random mode feature enables random selection of points. The filter can also generate vertices (topological primitives) as well as points. This is useful because vertices are rendered while points are not.

## 4.22 Outline

The Outline filter is a filter that generates a wireframe outline of any data set. The outline consists of the twelve edges of the data set bounding box.

## 4.23 Outline Corners

The Output Corners filter is similar to the Outline filter except that it generates lines only in the corners of the outline.

## 4.24 Point Data to Cell Data

The Point Data to Cell Data filter transforms point data (i.e., data specified per point) into cell data (i.e., data specified per cell). The method of transformation is based on

averaging the data values of all points defining a particular cell. Optionally, the input point data can be passed through to the output as well.

## 4.25 Probe

The Probe filter allows you to probe a data object at a point or along a line. When probing with a point you can interactively position the probe location and view attributes at this location. When probing with a line, a line interaction widget can be used to specify the line end points, and a plot of the attributes along this line is plotted. The Probe filter can be accessed through the Toolbar.

## 4.26 Quadric Clustering

The Quadric Clustering filter can be used to reduce the number of triangles in a triangle mesh, forming a good approximation to the original geometry. The input to `vtkQuadricClustering` is a `vtkPolyData` object, and all types of polygonal data are handled.

The algorithm used is the one described by Peter Lindstrom in his Siggraph 2000 paper, "Out-of-Core Simplification of Large Polygonal Models." The general approach of the algorithm is to cluster vertices in a uniform binning of space, accumulating the quadric of each triangle (pushed out to the triangles vertices) within each bin, and then determining an optimal position for a single vertex in a bin by using the accumulated quadric. In more detail, the algorithm first gets the bounds of the input poly data. It then breaks this bounding volume into a user-specified number of spatial bins. It then reads each triangle from the input and hashes its vertices into these bins. (If this is the first time a bin has been visited, initialize its quadric to the 0 matrix.) The algorithm computes the error quadric for this triangle and adds it to the existing quadric of the bin in which each vertex is contained. Then, if 2 or more vertices of the triangle fall in the same bin, the triangle is discarded. If the triangle is not discarded, it adds the triangle to the list of output triangles as a list of vertex identifiers. (There is one vertex id per bin.) After all the triangles have been read, the representative vertex for each bin is computed (an optimal location is found) using the quadric for that bin. This determines the spatial location of the vertices of each of the triangles in the output.

This filter can drastically affect topology, i.e., topology is not preserved.

## 4.27 Random Vectors

The Random Vectors filter assigns a random vector (i.e., magnitude and direction) to each point. The minimum and maximum speed values can be controlled by the user.

## 4.28 Reflection

The Reflection filter reflects a data set across one of the planes formed by the data set's bounding box or across one of the faces of the data set's bounding box or a plane at  $X=Center$ ,  $Y=Center$ , or  $Z=Center$  where Center is the value specified in the Center entry box. Since it converts data sets into unstructured grids, it is not efficient for structured data sets.

## 4.29 Ribbon

The Ribbon filter can be used to create oriented ribbons from lines defined in polygonal data set. The orientation of the ribbon is along the line segments and perpendicular to "projected" line normals. Projected line normals are the original line normals projected to be perpendicular to the local line segment. An offset angle can be specified to rotate the ribbon with respect to the normal.

## 4.30 Rotational Extrusion

The Rotational Extrusion filter is a modeling filter. It takes polygonal data as input and generates polygonal data as output. The input data set is swept around the z-axis to create new polygonal primitives. These primitives form a "skirt" or swept surface. For example, sweeping a line results in a cylindrical shell, and sweeping a circle creates a torus.

## 4.31 Shrink

The Shrink filter shrinks each cell of an arbitrary data set toward its centroid. The centroid of a cell is computed as the average position of the cell points. Shrinking results in disconnecting the cells from one another. The output of this filter is of type `vtkUnstructuredGrid`.

## 4.32 Smooth

The Smooth filter adjusts point coordinates using Laplacian smoothing. The effect is to "relax" the mesh, making the cells better shaped and the vertices more evenly distributed. Note that this filter operates on the lines, polygons, and triangle strips composing an instance of `vtkPolyData`. Vertex or poly-vertex cells are never modified.

The algorithm proceeds as follows. For each vertex  $v$ , a topological and geometric analysis is performed to determine which vertices are connected to  $v$  and which cells are connected to  $v$ . Then a connectivity array is constructed for each vertex. (The connectivity array is a list of lists of vertices that directly attach to each vertex.) Next, an iteration phase begins over all vertices. For each vertex  $v$ , the coordinates of  $v$  are modified according to an average of the connected vertices. (A relaxation factor is

available to control the amount of displacement of  $v$ ). The process repeats for each vertex. This pass over the list of vertices is a single iteration. Many iterations (generally around 20 or so) are repeated until the desired result is obtained.

## 4.33 Stream Tracer

The Stream Tracer filter will generate stream traces from a set of seed points. The seed points can be generated as random points within a sphere or evenly spaced points along a line. The center of the sphere and the end points of the line can be specified interactively. The input data object must have a vector field in order for this filter to work. The Stream Trace filter can be accessed through the Toolbar.

## 4.34 Threshold

The Threshold filter extracts cells that fall between a given lower and upper threshold. The threshold can be applied to cell scalars or point scalars. You can select whether all point scalars of a cell must meet fall within the threshold range in order to be included in the output or if just one falling within the range is sufficient. The Threshold filter is available on the Toolbar.

## 4.35 Triangle Strips

The Triangle Strips filter generates triangle strips and/or poly-lines from input triangles, triangle strips, and lines. Polygons other than triangles and lines are passed through to the output and are not stripped. (Use `vtkTriangleFilter` to triangulate non-triangular polygons prior to running this filter if you need to strip all the data.) The filter will pass through (to the output) vertices if they are present in the input polydata.

## 4.36 Subdivide

The Subdivision filter generates output by subdividing its input polydata. Each subdivision iteration creates four new triangles for each triangle in the input. This filter only operates on triangles; other cells are not considered during operation or passed through to the output. Use `vtkTriangleFilter` (Triangulate) first to create triangles from non-triangular data.

## 4.37 Tetrahedralize

The Tetrahedralize filter generates n-dimensional simplices from any input data set. That is, 3D cells are converted to tetrahedral meshes, 2D cells to triangles, and so on. The triangulation is guaranteed compatible as long as the dataset is either zero-, one- or two-dimensional; it works for a three-dimensional data set if all cells in the 3D data set are convex with planar facets.

## 4.38 Triangulate

The Triangulate filter generates triangles from input polygons and triangle strips. If requested, the filter also will pass through vertices and lines.

## 4.39 Tube

The Tube filter is a filter that generates a tube around each input line. The tubes are made up of triangle strips and rotate around the tube with the rotation of the line normals. (If no normals are present, they are computed automatically.) The radius of the tube can be set to vary with scalar or vector value. If the radius varies with scalar value the radius is linearly adjusted. If the radius varies with vector value, a mass flux preserving variation is used. The number of sides for the tube also can be specified, and the user has the option of whether to cap the tube.

## 4.40 Warp (scalar)

`vtkWarpScalar` is a filter that modifies point coordinates by moving points along point normals by an amount equal to the scalar value at that point times the scale factor. This is useful for creating carpet or x-y-z plots.

If normals are not present in the data, the Normal instance variable will be used as the direction along which to warp the geometry. If normals are present but you would like to use the Normal instance variable, set the Use normal boolean to true.

If the X-Y plane boolean is set true, then the z-value is considered to be a scalar value (still scaled by scale factor), and the displacement is along the z-axis. If scalars are also present, these are copied through and can be used to color the surface.

Note that the filter passes both its point data and cell data to its output, except for normals, since these are distorted by the warping.

## 4.41 Warp (vector)

The Warp (vector) filter moves data points along a vector direction. After specifying the vector field to use and the displacement distance, a new data object will be created by displacing each point by the specified distance along the specified vector. The Warp (vector) filter is accessible from the Toolbar.

# 5 Selection and Navigation

## 5.1 Overview

The Selection / Navigation Window is displayed on the top portion of the Left Panel when data has been loaded or created in ParaView, the Left Panel is not hidden, and a

Source or Filter property sheet is currently displayed in the Left Panel. The Selection / Navigation Window will appear similar to the example shown on the right.

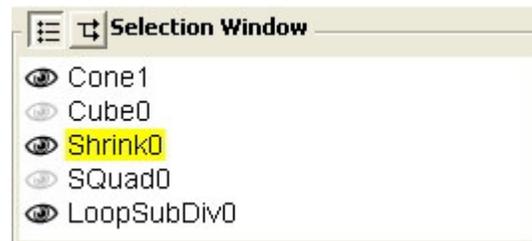


The title of the Selection / Navigation Window and the status of the two radio buttons indicate the current mode. In the example above, the first radio button (showing a list) is depressed and the title of the window is Selection Window. The Selection Window presents a list of all data objects in ParaView, allowing you to select the active one and view / control the visibility of each data object.

If the second radio button (showing a pipeline diagram) is depressed, the title will be Navigation Window. The Navigation Window presents a piece of the pipeline diagram for the current data objects and allows you to select the active one. These two types of windows are covered in more detail in the next two sections.

## 5.2 Selection Window

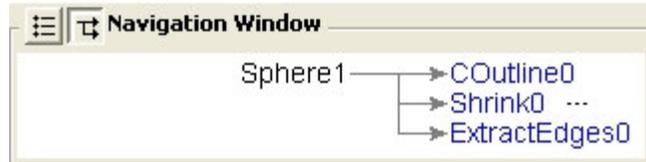
The first icon button in the Selection / Navigation Window is used to set the mode to Selection Window. In the image on the right you can see that this button, which looks like a list, is depressed, and the title of the window is Selection Window. In this mode you will see a list of data objects and an eye icon to the left indicating the visibility of the object. When this eye is light gray, the visibility of the item has been turned off. When the eye is black, the visibility is on. This does not necessarily mean that you will be able to see that item in the display area – it may be out of the current viewing frustum or hidden by an object in front of it.



Clicking on an item in the list will make it the currently active data object. You will see that item highlighted in yellow. If you create a filter it will be applied to this data object. In addition, the property sheet for this item will be displayed below the Selection / Navigation Window. Keep in mind that an item in the list represents the object used to read / create it (the reader, the source, or the filter), the data object created by that object (the output of the reader, source, or filter), and the objects involved in displaying the data (the mapper, the actor, and the property). From the property sheet you can access parameters of each of these objects associated with that list item. For example, selecting Cone1 from the list will display the property sheet with the cone source parameters on the Parameters tab; information about the output of the cone source on the Information tab; and properties of the actor, mapper, and property objects used to display the data (e.g., position, orientation, color, representation style, shading type, and color map options) on the Display tab.

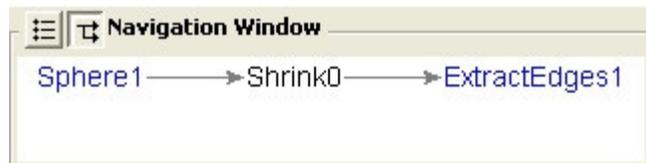
## 5.3 Navigation Window

The second icon button in the Selection / Navigation Window is used to set the mode to Navigation Window. In the image on the right you can see that this button, which looks like a small graph diagram, is depressed, and the title of the window is Navigation Window. A graphical



representation of the portion of the pipeline diagram directly connected to the selected item will be displayed. In the example above, three filters have been applied to a sphere source. The Sphere1 item is shown in black, indicating that it is the currently active item. The three objects connected to the output of the sphere source are shown in blue, indicating that you can click on the item to make it the currently active object and to

redraw the graph starting from this object. For example, clicking on the Shrink0 item in the above example leads to the pipeline diagram shown on the right. In this diagram we can see that the Shrink0 item gets its input from the Sphere1 source, and the output of the shrink filter is



the input to ExtractEdges1. Since three dots are used to indicate a continuation of the pipeline past the one item shown on either side, we can tell that Sphere1 is the beginning of this pipeline and ExtractEdges1 is the end of this pipeline since there are no dots. In the top example, note that there are dots after Shrink0 indicating a continuation.

Note that it is possible to access any data object using the Navigation Window, but it may require many steps since you can only move forward or backward one step in the pipeline at a time. For example, if Shrink0 is the active item as it is in the example above, then to make ExtractEdges0 the active item we would need to click on Sphere1 to go back a step, then click on ExtractEdges0 to go forward a step. Alternatively, you can toggle the mode to Selection Window, click on ExtractEdges0, then toggle the mode back to Navigation Window.

## 6 Toolbar

### 6.1 Overview

Directly below the main Menu Bar of ParaView is the Toolbar, providing access to some of the most common operations. An example of the Toolbar is shown below. The appearance of the Toolbar will change based on Toolbar Settings that can be found on the Application Settings property sheet. In the below example these settings indicate a flat frame and flat buttons.



Figure 8. ParaView Toolbar

Based on the current state of ParaView (i.e., the type of the current data object if at least one data object has been loaded or created) some of these buttons may be disabled. For example in the above image the last filter button is grayed out, indicating that it is disabled. This filter extracts a subgrid from structured data, and the currently active data object is polygonal (an unstructured data type).

The first three buttons in the toolbar control some common camera operations and are covered in the next section. The next ten buttons represent some common filters, and are covered in the Toolbar Filters section. The final four buttons control the center of rotation and are covered in the last section of this chapter.

## 6.2 Camera Controls

The first three buttons in the toolbar, as shown on the right, control the camera. The first button resets the view. The other two can be used to switch between 3D and 2D interaction modes. These are described in more detail below.



The reset button can be used to reset the camera so that all visible items will be in the viewing frustum. The current viewing direction will be maintained, but the camera and focal point will be translated, and the zoom factor will be adjusted so that all visible objects are in the view frustum. Items may still be hidden by another object nearer to the camera.



The 3D interaction mode can be selected using the second button in the Toolbar. Generally the difference between 2D and 3D interaction is that 3D interaction allows rotation while 2D interaction allows only translation (panning). A rotation arrow is used to indicate the 3D interaction mode. Details of the 3D interaction mode can be found in the Camera Controls section of the 3D View Properties chapter.



The 2D interaction mode can be selected using the third button in the Toolbar. An XY translation arrow is used to indicate the 2D interaction mode. Details of the 2D interaction mode can be found in the Camera Controls section of the 3D View Properties chapter.

## 6.3 Toolbar Filters

The second group of buttons contains ten commonly used filters. Some of these filters can only be applied to certain types of data, and therefore not all buttons will be enabled at all times. Filters that cannot be applied because the active data object is an incompatible input will have the corresponding shortcut button disabled and will appear grayed out. When the active data object is polygonal and has point scalars, the filter shortcuts on the Toolbar will appear as shown below.



These ten filters are briefly described here. More detailed descriptions can be found in the Filters chapter of this guide.



The Calculator filter can be used to create new arrays by performing mathematical operations on existing arrays. Using an interface similar to a calculator you can specify an equation to be applied to specified data array of the input data object.



The Warp Vector filter will move data points along a vector direction. After specifying the vector field to use and the displacement distance, a new data object will be created by displacing each point by the specified distance along the specified vector.



The Cut filter can be used to cut a data object with a plane or a sphere. This will usually decrease the dimensionality of the data object being cut. For example, cutting a sphere with a plane may produce a circle. An interactive widget can be used for placing the plane or sphere, and a set of offset distances can be specified to perform more than one cut at a time.



The Clip filter will clip a data object using a plane, a sphere, a box, or scalars. This will not decrease the dimensionality of the data object being clipped. For example, clipping a sphere with a plane may produce a hemisphere.



The Threshold filter will extract cells that fall between a given lower and upper threshold. The threshold can be applied to cell scalars or point scalars. You can select whether all point scalars of a cell must meet fall within the threshold range in order to be included in the output, or if just one is sufficient.



The Contour filter can be used to generate isosurfaces (from three dimensional data) or isolines (from two dimensional data). A set of scalar values can be specified to extract more than one contour at a time. New normals, gradients, and scalars can be optionally generated.



The Glyph filter will create a glyph at each point in the input data object. The glyph can be a cone, a sphere, a line, an arrow, or a 2D glyph. The glyph can be oriented according to a vector in the input data object and scaled according to a scalar, the vector magnitude, or vector components.



The Probe filter allows you to probe a data object at a point or along a line. When probing with a point you can interactively position the probe location and view attributes at this location. When probing with a line, a line interaction widget can be used to specify the line end points, and a plot of the attributes along this line is displayed.



The Stream Trace filter will generate stream traces from a set of seed points. The seed points can be generated as random points within a sphere or as evenly spaced points along a line. The center of the sphere and the end points of the line can be specified interactively. The input data object must have a vector field in order for this filter to work.



The Extract VOI filter allows you to extract a sub-grid from an input structured data object. The minimum and maximum extents of the volume of interest in X,

Y, and Z can be specified along with a sampling rate. Sampling rates greater than one will subsample the data.

## 6.4 Center Of Rotation Controls

The last four buttons in the Toolbar can be used to control the center of rotation. The appearance of this region of the Toolbar will depend on the state of the buttons. One possible configuration is shown below.



Using these buttons you can pick or reset the center of rotation, show or hide the marker, and edit the center of rotation coordinate values manually. These operations are explained below.



This button can be used to interactively pick the center of rotation. After pressing this button, the next mouse click in the Display Area will pick the new center of rotation. The location of the center of rotation will be based on the data found under the mouse position.



This button can be used to reset the center of rotation to be the center of the currently active data object. This is a useful operation to perform when you want to rotate around a particular data object to view it from all directions.



This button has two possible appearances based on the mode. When the button looks like the one shown on the left (with a dot in the center), the center of rotation marker is currently visible as red, green, and yellow lines that cross at the center of rotation coordinate location. Pressing this button will cause two things to happen. First, the center of rotation marker will become invisible. Second, this button's appearance will change to match the one shown in the Toolbar at the beginning of this section (without a dot in the center). Pressing the button now will once again show the center of rotation marker and toggle the appearance of this button.



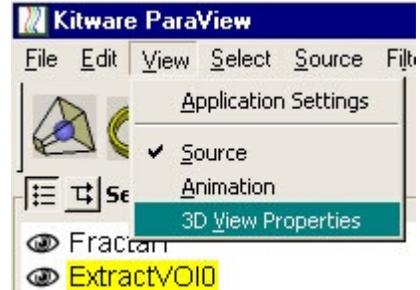
This button can be used to manually edit the center of rotation. By default it will appear as shown in the Toolbar above with a small right facing arrow in the corner. Pressing this button will lead to the interface seen on the right where the current center of rotation coordinate is displayed and can be edited using the text entry boxes. The button now has a left facing arrow, and when pressed will hide the text entry boxes again.

## 7 3D View Properties

### 7.1 Overview

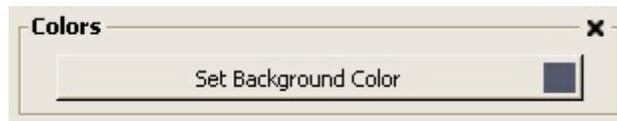
The 3D View Properties option under the View menu on the main Menu Bar can be used to access the property sheet that contains the user interface for controlling the camera, setting the background color, setting advanced rendering and level-of-detail

parameters, adding text annotation to the Display Area, and controlling the orientation axes. This property sheet contains three tabs: General, Annotate, and Camera. The next four sections cover the controls found on the General tab; the following two sections cover the Annotate tab; and the final four sections describe the parameters available on the Camera tab.

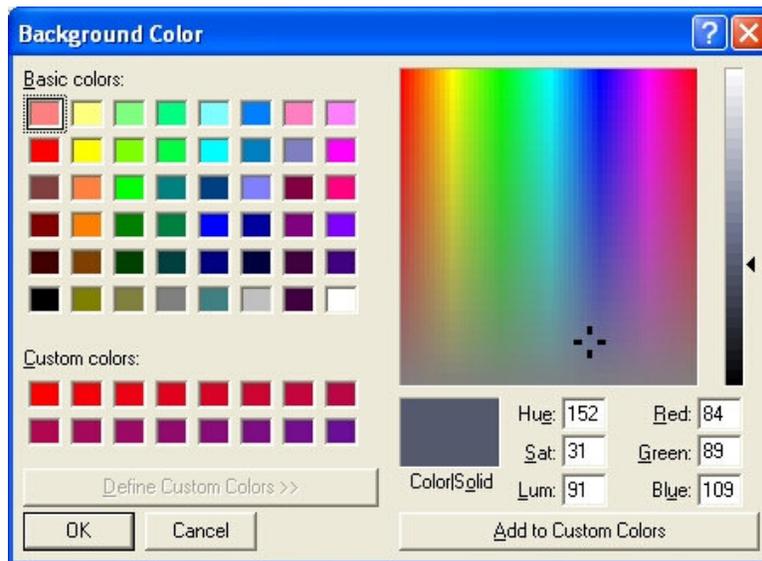


## 7.2 Background Color

A button for selecting the background color of the Display Area is provided on the General Tab of the 3D View Properties property sheet. This area of the interface appears as shown below.



Press the button to display a dialog from which you can specify a new background color. This dialog is platform specific, and may appear similar to the one shown below on Windows platforms.



When a new background color is selected, this information is saved in a ParaView registry file. This registry file is consulted on start-up; therefore the background color you select will remain the same even if you close ParaView and restart it. Changing the background to white is often useful before printing an image.

## 7.3 Advanced Render Parameters

A set of advanced rendering options are provided on the General tab of the 3D View Properties property sheet. This area of the interface appears as shown on the right. The

three check boxes can be used to control global properties of rendering such as parallel versus perspective projection, the use of triangle strips, and immediate mode rendering versus the use of display lists. These options are described below.



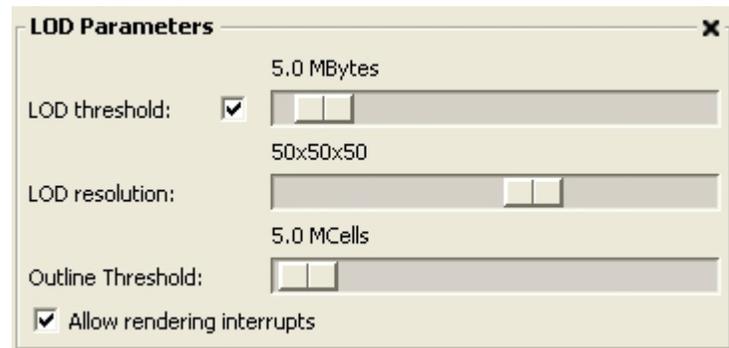
**Use parallel projection:** When checked a parallel camera projection matrix is used for rendering. When a parallel (orthographic) projection is used, objects do not appear smaller as they move farther away. When this option is not checked, a perspective camera projection is used which is more similar to the way our eyes see the world.

**Use triangle strips:** When checked all triangles will be converted to triangle strips before rendering. This may improve your rendering performance since less information needs to be passed to the graphics board with triangle strips. The image rendered with triangle strips (i.e., box is checked) will not necessarily be the same as the one rendered with triangles (i.e., box is unchecked), especially when flat shading is employed.

**Use immediate mode rendering:** When checked no display lists will be generated, and the data object will be traversed and rendered for each frame. When unchecked, a display list will be generated for the primitives. When an object does not change frequently, using display lists generally increases performance, but the initial creation of the display list can be slow on some platforms.

## 7.4 LOD Parameters

The controls for the levels of detail used to enable interactive rendering of large data are available in the LOD Parameters area of the General tab of the 3D View Properties property sheet. For large data sets that may require several seconds to render it is helpful to have a simplified version of the data that can be manipulated interactively, while still using the full resolution data to generate an image once interaction is complete. ParaView will automatically create the lower resolution representation of your data objects to ensure interactivity. The LOD Parameters area appears similar to the example shown above. Using these controls you can adjust the threshold at which an LOD model is created and the resolution of this LOD model. In addition, you can enable or disable rendering interrupts. Each of these features is described below.



**LOD threshold:** This slider can be used to control the threshold at which a lower resolution representation of your data is created. For data sets that are smaller than the

specified memory size, the original data will always be used for rendering. Moving the slider to the right will increase the threshold and will tend to render every data object at full resolution. Moving the slider to the left will lower the threshold and will give you better interactive performance by rendering lower resolution data.

**LOD resolution:** When a lower resolution version of the data is created, a 3D grid is used as part of a quadric clustering algorithm. This slider controls the resolution of this grid. Moving the slider to the left will lead to better approximations, while moving the slider to the right will produce coarser approximations with better interactive performance.

**Outline Threshold:** To allow for more interactive rendering rates, unstructured grid data sets larger than a certain size are rendered as an axis-aligned bounding box of the data set by default. (The representation can be changed to Surface on the Display tab of the property sheet for that data object.) The Outline Threshold slider determines how large (specified in millions of cells) an unstructured grid must be before it defaults to being drawn as an outline. The default is 5 million cells. Moving the slider to the left causes smaller data sets to be drawn as outlines; moving it to the right allows the surface of larger data sets to be drawn by default, but rendering the surface of large unstructured grids can lead to slower rendering times.

**Allow rendering interrupts:** When checked the rendering process in the Display Area can be interrupted by a mouse click. If you are rendering large data you may be able to interact with the data at a good rate due to the lower resolution version of the data, but it may require several seconds to refresh to the full resolution version when interaction is complete (the mouse button is released). If this button is not checked, then you must wait for the high resolution render to complete before further interaction can occur. When this button is checked, if you click in the Display Area to begin a new interaction or click on the interface to change the state of a widget, the current render will be aborted to respond. When MPI is in use, ParaView uses asynchronous messaging to provide this functionality.

## 7.5 3D Interface Settings

The 3D Interface Settings control area is found on the General tab of the 3D View Properties property sheet available under the View menu on the main Menu Bar.

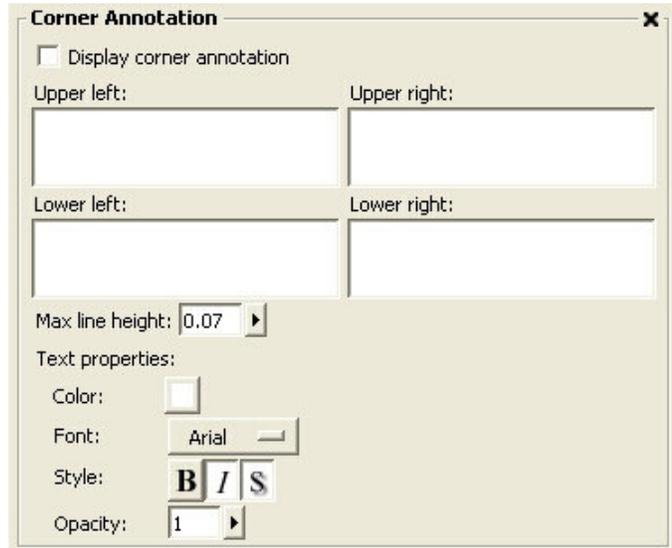
This area of the interface appears similar to the example shown on the right. There is one configurable parameter in this region as described below.



**Display 3D widgets automatically:** This option is checked by default indicating that 3D interaction widgets will be automatically displayed when creating a filter that utilizes such a widget for parameter specification. If you are working with very large data and plan to specify all parameters manually, unchecking this option allows you to avoid the extra render that occurs from turning on the 3D widget.

## 7.6 Corner Annotation

Text annotation can be added to the four corners of the Display Area using the Corner Annotation area on the Annotation tab of the 3D View Properties property sheet. This property sheet is displayed when 3D View Properties is selected from the View menu on the main Menu Bar. This area of the interface appears similar to the example shown on the right. Using these controls you can enter text for each of the four corners, control properties of the text such as color and font, and toggle the visibility of the annotation. Each of the parameters is described in more detail below.



**Display corner annotation:** When checked the corner annotation is visible.

**Upper left, Lower left, Upper right, Lower right:** These four text entry boxes can be used to add multi-line text to the specified corner. The text entered will automatically scale to fit the available space. The longer the string and the more lines of text you specify, the smaller the text size.

**Max line height:** Set the maximum height of a line of text. This value is a percentage of the total amount of vertical space allocated to this corner text. The height may actually be smaller than this value if this is necessary to fit all of the text.

**Color:** The first of the Text properties buttons can be used to specify the color of the text.

**Font:** The second Text properties button is used to determine the font family.

**Style:** These three buttons control whether or not the text is bold, italicized, or shadowed. When one of these buttons appears depressed the option is enabled; when the button is raised the option is disabled. In the above example image, the bold option is disabled, and the italics and shadow options are enabled.

**Opacity:** The final Text properties control is for the opacity of the text and can range from 0.0 (transparent) to 1.0 (opaque).

## 7.7 Orientation Axes

The Orientation Axes area on the Annotation tab of the 3D View Properties property sheet (displayed when 3D View Properties is selected from the View menu on the main Menu Bar) can be used to control the behavior of the orientation axes widget. (The

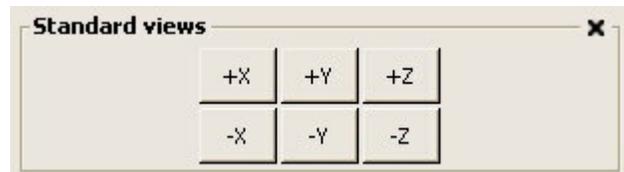
orientation axes widget displays the current orientation of the scene.) This area of the interface appears similar to the example shown on the right.



The first check box, Display orientation axes, determines whether the orientation axes 3D widget is visible. The second check box, Interactive, determines whether the orientation axes can be positioned and sized interactively in the Display Area. If Interactive mode is on, the Set Outline Color button is used for setting the color of the outline of the orientation axes widget. The outline makes it easier to find the edges of the 3D widget to resize it.

## 7.8 Standard Views

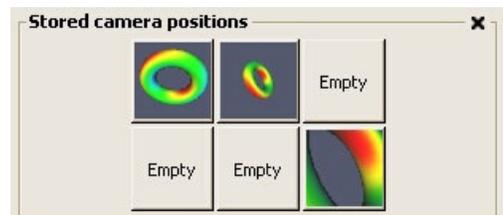
The Standard Views area can be used to reset the camera to view the scene from a particular viewing direction. This area is located on the Camera tab of the 3D View Properties property sheet, which can be accessed through the View menu on the main Menu Bar. The Standard Views area contains six buttons as shown on the right which can be used to reset the camera so that all of the visible data objects are within the viewing frustum, and the camera direction of projection lies along the indicated axis. This is different than the reset camera button located in the Toolbar which does not change the direction of projection.



The Standard Views area contains six buttons as shown on the right which can be used to reset the camera so that all of the visible data objects are within the viewing frustum, and the camera direction of projection lies along the indicated axis. This is different than the reset camera button located in the Toolbar which does not change the direction of projection.

## 7.9 Stored Camera Positions

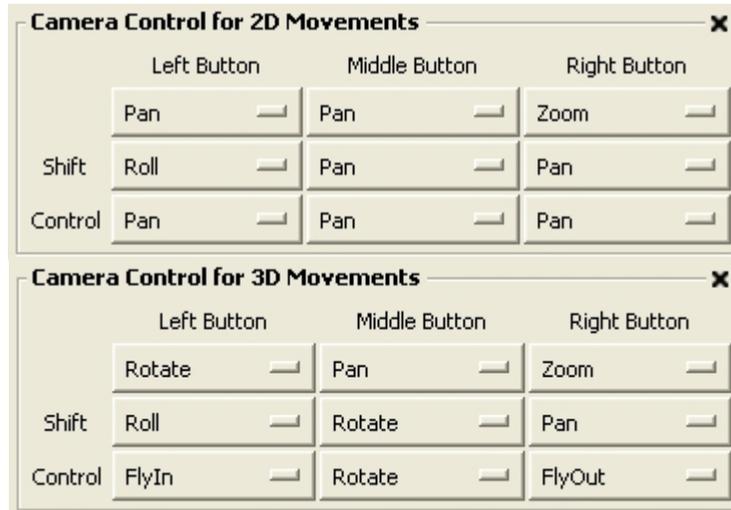
The Stored Camera Positions area can be used to store and retrieve up to six camera viewing positions and is located on the Camera tab of the 3D View Properties property sheet. This property sheet can be accessed through the View menu on the main Menu Bar. In the example shown on the right you can see that three camera positions have been saved, and three buttons are still empty. To store the current camera parameters in one of the six locations, right click on that button. To retrieve the saved position, left click on the button.



## 7.102D / 3D Camera Controls

There are two types of camera interaction modes available in ParaView – 2D interaction and 3D interaction. These two modes allow you to customize the mouse operations for how you want to interact in 2D (perhaps when viewing an image) versus how you would like to interact with objects in 3D and to choose between these two modes during a ParaView session. The control for switching between the two modes is provided with the second two buttons on the Toolbar.

Two interface regions are provided on the Camera tab of the 3D View Properties property sheet for customizing the mouse interaction in each of the possible modes. You can access this area of the interface by selecting 3D View Properties from the View menu on the main Menu Bar. The 2D and 3D regions appear similar to those shown below.



The Camera Control for 2D Movements area contains a subset of the possible interaction styles that can be associated with key/mouse combinations in the 3D movements area. All of the interaction styles are described below.

**Pan:** The mouse motion will translate the camera in the view plane (X and Y axes of the camera coordinate system). Panning is available for both 2D and 3D modes.

**Roll:** The mouse motion will roll the camera. This form of rotation is available in both 2D and 3D modes.

**Zoom:** Zoom in or out of the image. In a parallel projection this is a change in the parallel scale, while in a perspective projection this is a change in the field of view. Zooming is available in both 2D and 3D modes.

**Rotate:** Rotate the camera around the center of rotation using azimuth and elevation operations. The rotate function is available only in 3D mode.

**FlyIn:** Move the camera in the direction indicated by the mouse. Rotation is controlled by the placement of the mouse. The farther from the center of the Display Area, the faster the rotation. Speed is decreased as rotation occurs. Unlike other operations that require mouse movement to cause a change in the camera parameters, the FlyIn and FlyOut operations will execute continually while the corresponding key/mouse buttons are pressed. FlyIn is not available in 2D interaction mode.

**FlyOut:** Move the camera away from the direction indicated by the mouse. This option can be used to "back up" after flying in towards an object. FlyOut is not available in 2D interaction mode.

**Move:** The data object under the mouse cursor when the specified key/mouse combination is pressed will be translated according to the mouse motion. This is an

interactive method for placing objects in the scene. Unlike other operations, the Move interaction style changes the position of the data object instead of the position of the camera.

## 7.11 Camera Control

In addition to controlling how the camera behaves because of mouse movement, the user can also directly change the elevation, azimuth, and roll of the camera. The interface for doing this is



found on the Camera tab of the 3D View Properties property sheet, available through the View menu on the main Menu Bar. This interface appears similar to the one on the right. Pressing any one of the buttons, Apply Elevation, Apply Azimuth, and Apply Roll, moves the camera in the specified manner by the number of degrees in the entry box to the right of the button pressed. Pressing one of these buttons repeatedly will incrementally move the camera by the specified number of degrees (e.g., if you enter 30 in the entry box beside the Apply Elevation button, the first time you press the button the elevation will change by 30 degrees, and the second time it will change by 30 more degrees – a total of 60 degrees).

# 8 Display Properties

## 8.1 Overview

When a data object is created by loading data, creating a source, or applying a filter, a property sheet is associated with that data object. To view this property sheet, select the data object from the list of objects in the Select menu on the main Menu Bar, or select the item from the Selection / Navigation Window. This will cause the property sheet to be displayed in the Left Panel below the Selection / Navigation Window.

The property sheet associated with the item has three tabs – one marked Parameters that contains parameters for the reader, source or filter that generated this data, one marked Display which controls properties which affect the visualization of the data, and one marked Information with general information about the data object. This chapter is dedicated to the Display tab of this property sheet.

## 8.2 View Settings

The top region on the Display tab of the property sheet associated with an item in ParaView contains the View settings.



This region of the interface allows you to toggle the visibility of the data and other items associated with the data and to reset the camera to view this data object. Each of these settings and operations is described in more detail below.

**Data:** This check box can be used to toggle the visibility of the data. This visibility can also be controlled using the eye icon in the Selection Window. This check box and the eye icon are synchronized.

**Scalar bar:** If the Color by option in the Color area is set to use a scalar or vector component for color, then the Scalar bar option will be enabled in the View area allowing you to toggle the visibility of the scalar bar. The scalar bar provides a graphical representation of the color map used to color the image, along with tick marks indicating the range of values mapped through the table. The scalar bar is an interaction widget – using the left mouse button, you can reposition and resize the scalar bar in the Display Area. Bringing the scalar bar close to the bottom or top edge of the Display Area will cause it to switch to a horizontal mode while bringing it near the right or left edge will cause it to switch to a vertical mode.

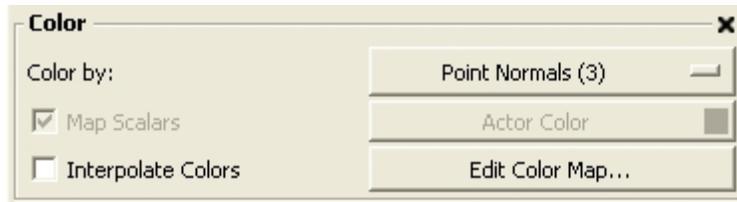
**Cube Axes:** Toggle the visibility of the labeled axes displayed along three outer edges of the bounding box of the data object.

**Label Point Ids:** If Label Point Ids is selected, point ids for the current data set will be drawn in the Display Area. The option of labeling point ids is only available when running ParaView in a single process.

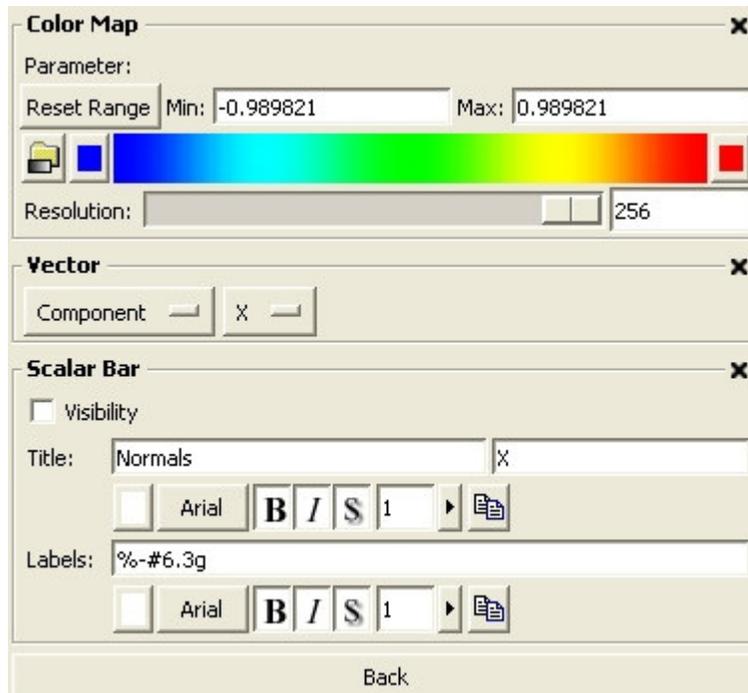
**Set View to Data:** This button will reset the camera so that this data object mostly fills the Display Area. The current camera direction will be maintained, but the camera will be translated and the zoom will be adjusted.

## 8.3 Color Options

The Display tab on the property sheet associated with a data object in ParaView has a region for adjusting the color of the actor as shown below. In the example shown, the Color by selection has been set to Property, and an Actor Color button is displayed to allow you to choose a new solid color for this actor. The dialog box for selecting a new color is operating system dependent. On Windows, the dialog box will look similar to the one shown in section 7.2 (Background Color). The Color by setting can instead be set to any scalar or vector component of this data object. When the setting is not Property, then the Actor Color button is disabled, and the Edit Color Map option is enabled.



**Edit Color Map...:** If the Color by option in the Color area is set to use a scalar, a vector component, or vector magnitude for color, then this button will be enabled. Pressing this button will change the property sheet to look similar to the image shown below.



The top region of this property sheet allows you to adjust the color map used to map scalars, vector components, or vector magnitudes to colors. The indicated range is the initial range computed for the array among all data objects. If you have adjusted parameters that have changed the scalar range you will either need to press Reset Range to reset to the full scalar range or enter a sub range manually. There are several preset color maps, or you can specify the color of the two end points of the color map. An HSV interpolation method will be used between these end points. The resolution of the color map can be set to any value between 2 and 256. This number indicates how many colors are in the map.

When the array used to color by is a vector, the Vector region appears. You can select a component or magnitude to color by using these dropdown menus.

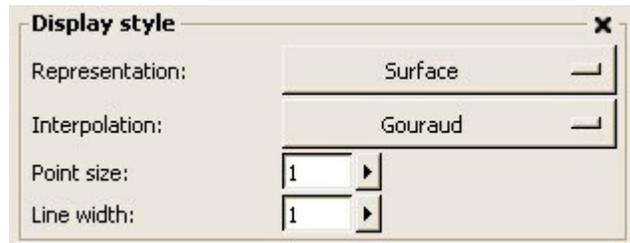
The bottom region of this property sheet contains controls for customizing the scalar bar. You can toggle the visibility, change the title, and adjust the formatting for the tick labels. In addition, you can set the color, font family, attributes (bold, italic, and shadow), and the opacity of each text item. For convenience, the last button in the row will copy the properties from the other item. For example, if you customize the Title text then want

to copy the specification of this text to the labels, press the  button at the end of the text customization buttons in the Labels area.

Pressing the Back button will return you to the Display tab of the data object's property sheet.

## 8.4 Display Style

On the Display tab of the property sheet associated with a data object in ParaView you will find a Display style region similar to the one shown in the image on the right. In this area you can change the representation of the data, the interpolation method, and the size of points and lines as described below.



**Representation:** You can choose between a surface, a wireframe, an outline, or a point representation of the data. When a surface representation is selected, primitives such as polygons and triangles are drawn as filled geometric primitives. When wireframe mode is selected, only the edges are drawn. In outline mode, only the bounding box of the data set is shown. In point mode, only the points defining the geometry are displayed. If this data object is an unstructured grid, a volume rendering option is also available from this menu.

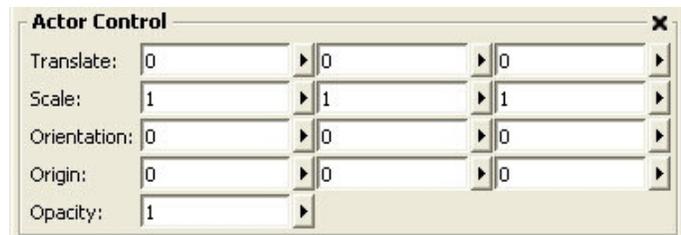
**Interpolation:** In Gouraud interpolation the normal will be interpolated over the face of a polygon for shading. When flat shading is selected only one normal will be used per polygon leading to a more faceted appearance. Objects that do not have point normals will be drawn with flat shading regardless of the setting of the pulldown.

**Point size:** When displaying points either as part of the data object or due to a Representation of points, the point size can be controlled with this slider.

**Line width:** If lines are being drawn for the data object (either because lines are a part of the data object or because the representation has been set to wireframe or outline), this slider can be used to control the thickness of the lines.

## 8.5 Actor Control

The Actor Control region of the Display tab on the property sheet associated with a data object can be used to adjust the position, scale, orientation, origin, and opacity of the actor associated with this data. This region of the interface will appear



similar to what is shown in the image on the right. Each of the input areas is covered in more detail below. In each of the text entry areas, the action will not occur until the enter

key is pressed in the area or the text widget loses focus (for example when you click in another text entry area). Using the sliders or thumbwheels causes the action to occur immediately.

**Translate:** These three values represent a translation in X, Y, and Z. Either enter these values manually, or change them with the associated thumb wheels.

**Scale:** Enter scale values along the X, Y, and Z axes, or set the values using the thumb wheels.

**Orientation:** Sliders are provided to adjust each of the scale entries to a value between 0 and 360 degrees. These represent rotations around the X, Y, and Z axis and are applied in the order Z, then X, then Y.

**Origin:** This specifies the origin of the data set. Either type the values in the entry boxes, or set the values using the thumb wheels.

**Opacity:** Use the slider to adjust the opacity of the data object to a value between 0.0 and 1.0. Keep in mind that polygons are not sorted by depth in ParaView, and therefore images generated with translucent polygons will often be incorrect.

## 9 Data Information

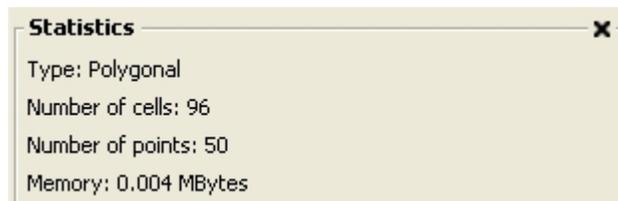
### 9.1 Overview

When a data object is created by loading data, creating a source, or applying a filter, a property sheet is associated with that data object. To view this property sheet, select the data object from the list of objects in the Select menu on the main Menu Bar, or select the item from the Selection / Navigation Window. This will cause the property sheet to be displayed in the Left Panel below the Selection / Navigation Window.

The property sheet associated with the item has three tabs - one marked Parameters that contains parameters for the reader, source or filter that generated this data, one marked Display which controls properties which affect the visualization of the data, and one marked Information with general information about the data object. This chapter is dedicated to the Information tab of this property sheet.

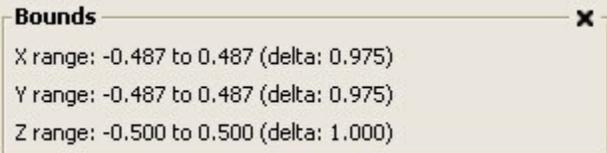
### 9.2 Statistics

The top region on the Information tab of the property sheet associated with a data object will provide basic statistics about that data object including the type of data, the number of cells, the number of points, and the size in memory. An example is shown on the right for polygonal (vtkPolyData) data.



## 9.3 Bounds

Below the Statistics region on the Information tab of the property sheet associated with a data object is a region for displaying bounds. An example is shown in the image on the right. The range on each axis is displayed both as the minimum and maximum coordinate in the data object on that axis and as a delta value between the two extremes.



## 9.4 Extents

For structured data, a third region will be displayed on the Information tab on the data property sheet associated with a data object. This region will show extent information as shown in the image on the right. The extent along each axis indicates the minimum and maximum index used to access the data along that axis. The dimensions are also given and are equal to the maximum index minus the minimum index plus one (since the indices are inclusive).



# 10 Saving Results

## 10.1 Overview

Several types of data can be saved from ParaView. The currently selected data object can be saved in several formats; a session file can be saved, capturing the state of the system; and an image of the Display Area can be saved.

## 10.2 Supported Data Formats

When the Save Data option is selected from the File menu, you will be able to save the currently active data object as a ParaView file (parallel or single, supports groups and multi-block), a VTK file (parallel or single, single block), a legacy VTK file (single process, single block), a Meta Image file (for image data only), or an Xdmf data file (if ParaView was compiled with Xdmf support).

## 10.3 ParaView Session Files

Selecting the Save Session State or Save Session Trace options from the File menu will save the state of the system either as it is at the moment of saving (Save Session State) or as a history of operations performed since ParaView was started (Save Session Trace). Reloading either of these will return ParaView to the state when the file was saved, but if sources or filters were both added and deleted, loading the trace file will

take longer (because these add and delete operations were recorded) than loading a corresponding state file.

## 10.4 Supported Image Formats

If you select the Save View Image from the file list you will be able to save the current image in the Display Area. Supported image file formats are Windows Bitmap (\*.bmp), JPEG (\*.jpg), PNG (\*.png), binary PPM (\*.ppm), and TIFF (\*.tif).

# 11 Copy and Print

## 11.1 Overview

Depending on which platform you are using, ParaView may include a copy image operation and printing functionality. The copy image option is supported only on the Windows platform. Printing is fully supported on Windows. On Linux / Unix, the print functionality will generate a postscript file that you must send to your printer.

## 11.2 Image Copy

To copy the current image in the Display Area, select the Copy View Image option from the Edit menu on the main Menu Bar. You can now past this image into another application using either the Edit | Paste menu option or Ctrl-v, as supported by the application. This functionality is available only on Windows platforms.

## 11.3 Image Print

The Print option in the File menu on the main Menu Bar can be used to print the current image in the Display Area. Before printing, you should select a DPI (dots per inch) setting from the Page Setup cascading menu under the File menu. Available options are 100, 150, and 300 DPI. The higher the DPI the better quality the printed image will have but the longer it will take to print.

On Windows, after you select the Print option a Windows dialog will pop up to select the printer. On Linux / Unix a file saving dialog will pop up allowing you to specify the location and name of a postscript file into which the image will be written. On Linux / Unix you will have to send this image to the printer.

# 12 New features

The new features added in the 1.6 release of ParaView are as follows.

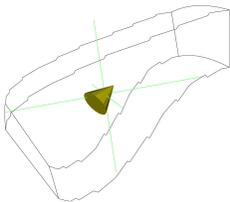
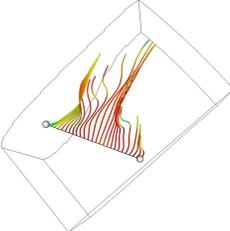
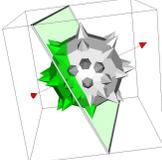
**Interpolating Colors:** A new check box labeled “Interpolate Colors” has been added to the Display tab of the data property sheet. Turning this option on causes the RGB (red, green, blue) color values to be interpolated in RGB space across polygons.

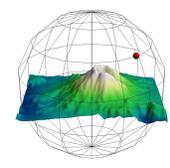
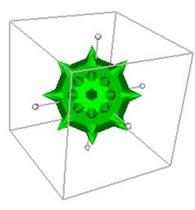
**Using Label from Parameters tab:** The user-specified Label value is now used in the Select menu and in the Animation interface in place of the system-assigned Name value.

**Volume Rendering:** For unstructured grid data sets with at least one point-centered scalar attribute, a volume rendering option is available from the Representation menu in the Display Style section of the Display tab of the data property sheet. From the Edit Volume Appearance... button (also on the Display tab), you can modify the transfer functions controlling the appearance of the unstructured grid.

## Appendix A: 3D Widgets

ParaView uses the following 3D widgets:

<p><b>Point Widget:</b> Used to set the position of a point or the center of a region. The position of the point can be changed by clicking on and dragging any of the three axes. To position the widget accurately, you might need to change the camera position as well. Holding the SHIFT key while interacting will restrict the motion of the point to one of the x, y or z planes.</p>	
<p><b>Line Widget:</b> Used to set the orientation and the position of a line. The position of the line can be changed by clicking on any point on the line except the end-points and dragging. To position the widget accurately, you might need to change the camera position as well. Holding the SHIFT key while interacting will restrict the motion to one of the x, y or z planes. To move one of the end-points, simply use one of the point widgets on each end. These are marked by spheres which become red when clicked on. Right-clicking and dragging the line will cause it to resize. Upward motion increases the length of the line; downward motion decreases it.</p>	
<p><b>Plane Widget:</b> Used in manipulating clipping and cutting planes. The plane can be moved parallel to its normal by clicking on any point on the plane except the line center and dragging. The plane normal can be changed by manipulating one of the point widgets at each end of the normal vector. Right-clicking and dragging the plane will cause it to resize. Upward motion increases the area of the plane; downward motion decreases it.</p>	

<p><b>Sphere Widget:</b> Used in clipping and cutting. The sphere can be moved by left clicking on any point on the sphere and dragging. The radius of the sphere is manipulated by right clicking on any point on the sphere and dragging.</p>	
<p><b>Box Widget:</b> Used in clipping and transforming. The box can be used to specify translation, scale, and orientation. Each face of the box can be positioned by moving the handle (sphere) on that face. Moving the handle at the center of the box causes the whole box to be moved. (This can also be achieved by holding the SHIFT key while interacting.) The box can be rotated by clicking and dragging with the left mouse button on a face (not at the handle) of the box. Clicking and dragging inside the box with the right mouse button uniformly scales the box.</p>	

## Appendix B: Tutorials

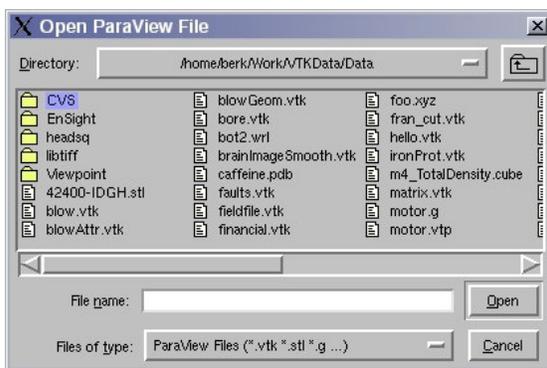
### Tutorial I: Visualizing Streamlines

This example will introduce more basic concepts including loading data files, interacting with 3D widgets, and saving data files. If you are not familiar with ParaView, we recommend that you read the first tutorial before working through this example.

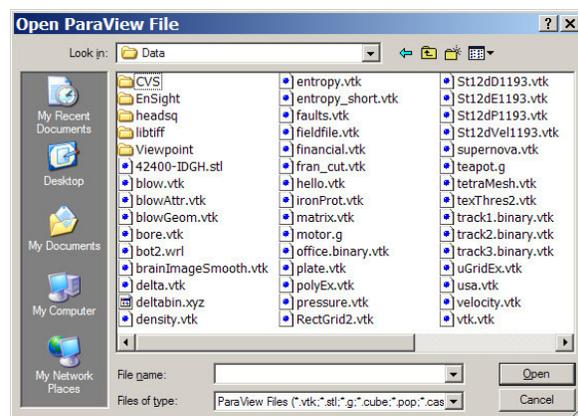
#### 1. Load the Data File

This tutorial assumes that you have not performed any actions before following the tutorial steps. Otherwise the names of the modules created might vary. For instructions on how to start ParaView, see section 1.2.2, Starting ParaView.

To load the data file you will use in this tutorial, select “File | Open Data” from the Menu Bar. This will launch one of the file selection dialogs shown below.



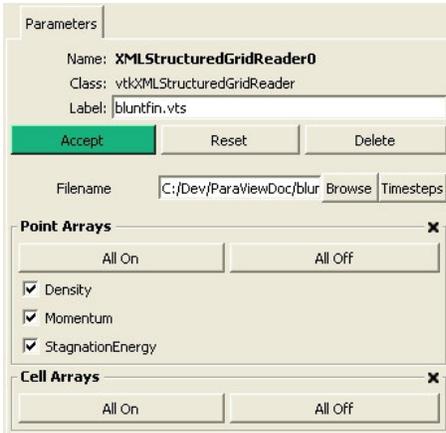
Unix and Linux



Windows XP

Figure 9. File Open Dialogs.

Initially, the file selection dialog shows all files with extensions that ParaView recognizes. You can limit the display to files of certain type by selecting a different filter using the “Files of type:” menu. See section 2.2, Supported Data Formats, for more information on file formats supported by ParaView.



Navigate to the folder in which the tutorial data files are located. (The tutorial data files are available in the Download section of the ParaView web site.) Next, select bluntfin.vts and click Open. (If the file is not listed, make sure that the tutorial data files have been downloaded and extracted and that the file selection dialog is showing all ParaView file types by selecting ParaView Files in the “Files of type:” menu.) The Left Panel should now look similar to the image shown on the left. Up to this point, ParaView read information about the contents of the data file but did not actually read any data. This allows the user to change certain properties of the reader before the file is read. As shown in the picture, the properties frame now contains a list of all the data arrays (variables, attributes) available in the file. In this case, the file contains two scalar variables called Density and StagnationEnergy and one vector variable called Momentum. By default, all arrays are selected and will be read if you click Accept. However, since you will need only the Momentum vector in this tutorial, you can reduce the load time and memory consumption by deselecting the Density and StagnationEnergy scalars. Note that only some readers support this feature. If a reader does not have any properties accessible to the user, it will immediately load the data file. After deselecting Density and StagnationEnergy, click Accept. The ParaView window should now resemble the image below.

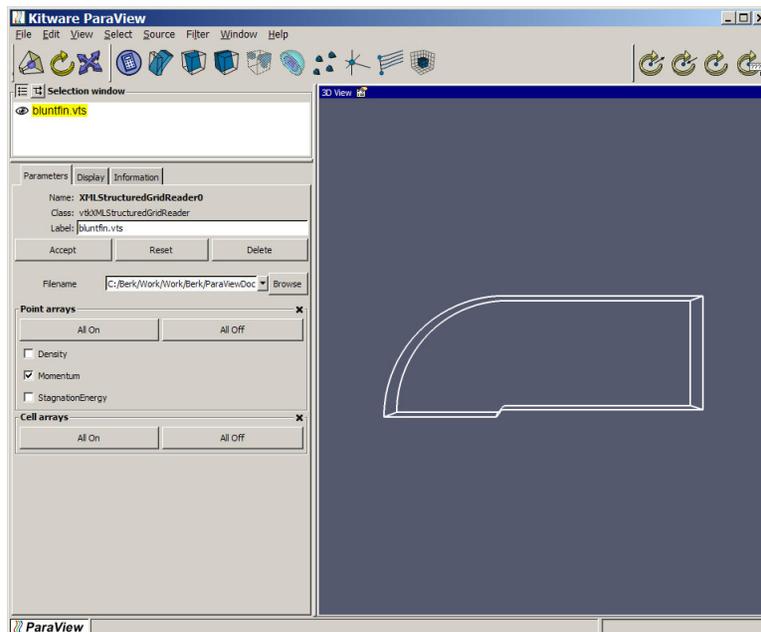


Figure 10. ParaView window after opening a structured grid data file.

The file you just download contains structured grid data showing the supersonic flow over a flat plate with a blunt fin rising from the plate. It was converted from a PLOT3D file distributed by NASA (<http://www.nas.nasa.gov/Research/Datasets/datasets.html>). You will notice that only a wireframe outline of the grid is displayed. ParaView displays structured datasets (rectilinear and curvilinear grid) and large unstructured grids as outlines and polygonal data and small unstructured grids as surfaces. For more information on data types supported by ParaView, see section 2.3, Data Types.

## 2. Extract Sub-grids



Next, we will extract two 2D sub-grids from the structured grid using the extract filter. Click on the Extract icon on toolbar to create the first filter. In the VOI (Volume of Interest) section of the property sheet shown on the Left Panel, set the minimum and the maximum values of I to 18 by either typing in the entry box or using the sliders. Click Accept. The output should look like the image below.

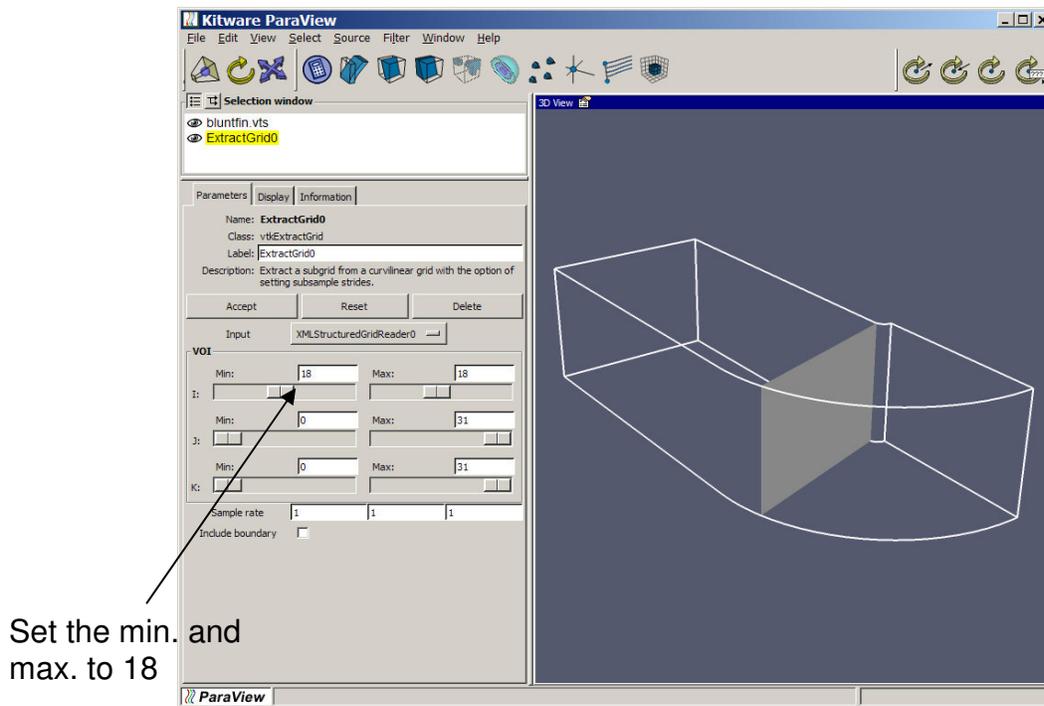


Figure 11. First sub-grid.

Extract is a filter that either selects a portion of a structured grid dataset or sub-samples it. The selected portion is referred to as the Volume of Interest (VOI). The sampling rate is determined by the “Sample Rate” parameter. Note that screen shot above was generated after changing the camera position by using the 3D interaction methods described in the previous tutorial. To change the color mapping of the new plane, select the Display page and choose “Point Momentum (3)” from the Color by menu. Next, we will extract the fin surface. First, make sure that the 3D structured grid is the active data. (Click on bluntfin.vts in the Selection Window.) Now create another extract filter by clicking on its icon on the toolbar. Set the minimum and the maximum of J (in the VOI

section) to 0 by either typing in the entry box or using the sliders. Click Accept. ParaView should now look like this.

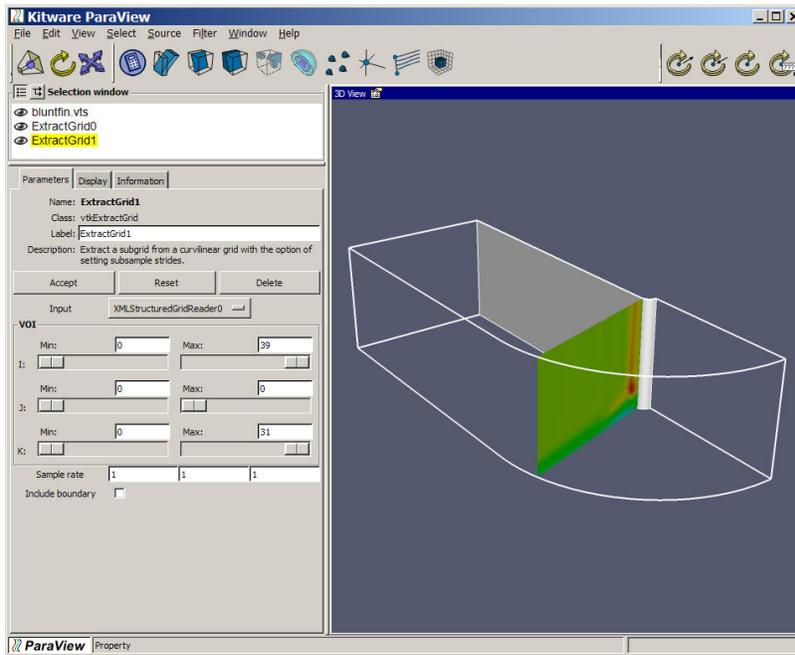


Figure 12. Second sub-grid.

### 3. Create Streamlines

Next, we will add some streamlines. First, make sure that the 3D structured grid is the active data. (Click on blunffin.vts in the Selection Window.) Then, create a Stream Tracer filter by clicking on the Stream Tracer icon on the toolbar. The Stream Tracer filter has more complicated properties than previously encountered. The most important of these is the seed property. This property has two components. The first one is displayed on the property page and allows you to modify the settings of the seed(s) textually. The second one is shown in the Display Area and allows you to modify the settings of the seed(s) interactively. To better see the point widget in the Display Area, you can turn off the visibility of the first sub-grid by clicking on the eye icon next to ExtractGrid0 in the Selection Window.

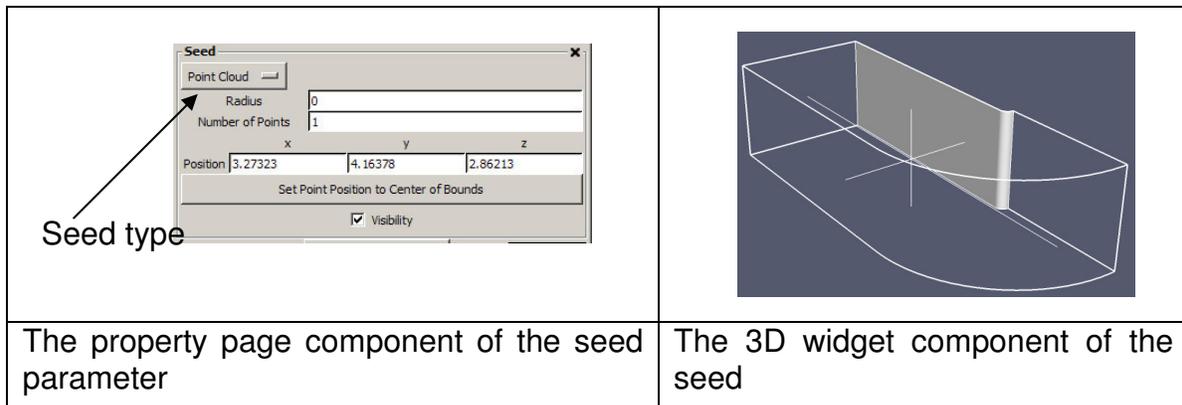
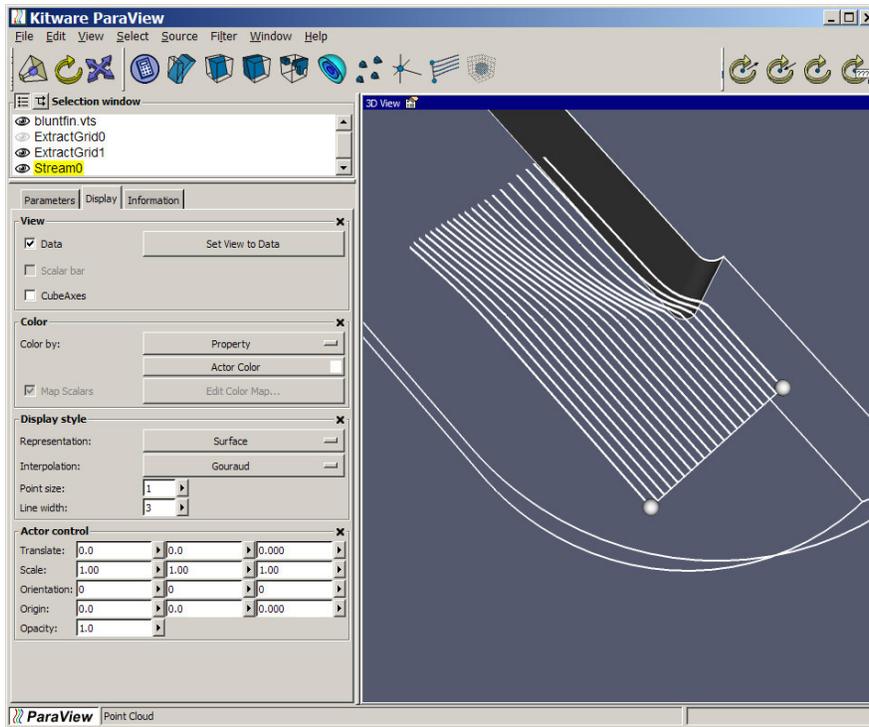


Figure 13. Components of the seed parameter.

ParaView supports a few manipulators (3D widgets) to allow users to easily change parameters. For a list of the 3D widgets used in ParaView, see Appendix A: 3D Widgets. You will first use a line to seed the streamlines. Select Line as the seed type from the pull-down menu in the Seed property frame. Next, set point 1 to (-3.59, 0.33, 2.86) and point 2 to (-3.59, 5.48, 2.86) by typing in the entry boxes. You can also change the position of the end points by clicking on the spheres at each end of the line and dragging. Next, set the maximum propagation length to 10 and set the integration direction to FORWARD. Click Accept. The output should look like the following.



**Figure 14. Streamlines created with a line seed.**

Next, we will move the seed to a more interesting region. First, change the seed type to Point Cloud, and change the Radius to 0.3 and the Number of Points to 10. Next, position the center of the point cloud at approximately (-0.54, 0.28, 0.55) by typing in the entry boxes. (You can also move the point widget by clicking on one of the axis and dragging.) This point is near the stagnation point on the plate-fin juncture. Click Accept. The Point Cloud seed source will create 10 points randomly placed in a sphere of radius 0.3 centered at the position of the point widget. To view the structure in the juncture region more closely, move the camera near the center of the point cloud. ParaView provides a few tools to make the placement of the camera in 3D easier.

#### 4. Add Tubes

Unshaded polylines are not the best way to visualize streamlines in 3D. Adding some thickness (and shading) can make it much easier to comprehend the three-dimensional structure of the flow field. This can be done by using either the tube or the ribbon filter. The tube filter generates tubes around input lines, whereas the ribbon filter generates oriented ribbons from the input lines. In this tutorial, we will use the tube filter. First,

make sure that the streamline filter's output is the active data. (Click on Stream0 in the Selection Window or choose Stream0 from the Select menu.) Next, create a tube filter by selecting "Tube" from the Filter menu. Set the "Number of Sides" parameter to 8 and Radius to 0.03. Click Accept. You can change the color mapping by selecting the Display tab in the properties page and selecting an array from the Color by menu. The ParaView window should now look like this.

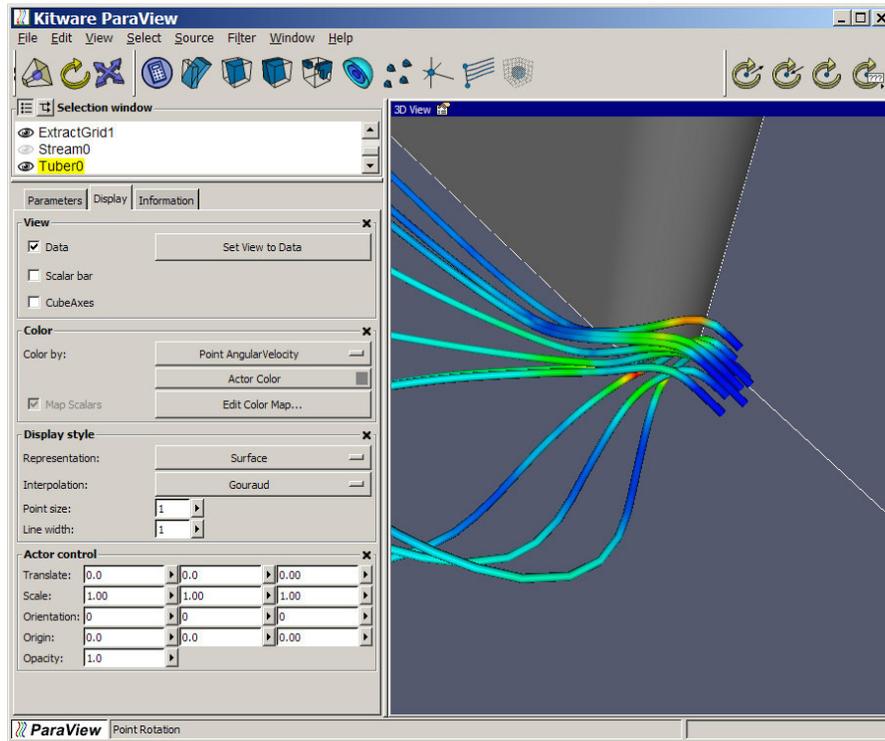


Figure 15. Tubed streamlines.

You might have noticed that the filters available in the Filter menu do not always remain the same. This is because the Filter menu is context sensitive and will only enable filters applicable to the current data object. For example, if the current object is of type structured grid, the tube filter will not be active in the Filter menu since it can be only applied to polylines. Similarly, if the current data object does not contain any three-component vectors, the streamline filter will not be available. Every time a new data object is created or selected, ParaView modifies the availability of the filters in the filter menu by comparing the requirements of each filter to the properties of the current object and adding only applicable filters.

## 5. Save the Tubes

Since the example data file used in this tutorial was small, the streamline calculation took very little time and can be easily repeated without any inconvenience. However, as the data files become larger, such calculations may take a long time. In such cases, it might be useful to store the result of the calculation so that it can be restored later and viewed in 3D. To save the output of the tube filter, select the Tuber0 object from the Selection Window (or the Select menu) and select File | Save Data. This will bring up a

Save Data selection window that is very similar to the Open Data dialog. Change the path to the directory where you want your data file(s) stored. In this case, you can write the data object in five different formats: ParaView Data format (\*.pvd), VTK polydata format (\*.vtp), legacy VTK format (\*.vtk), PLY Polygonal File Format (\*.ply), and XDMF format (\*.xmf).

This concludes the second tutorial. You can now exit ParaView by either selecting File | Exit or closing the window.

## Tutorial II: Animating Isosurfaces

This tutorial will introduce ParaView's animation interface. It will demonstrate the use of a few ParaView filters, including the isosurface, clip, and cut filters, and show how these can be controlled by the animation editor. If you are not familiar with ParaView, we recommend that you read the first two tutorials before working through this example.

### 1. Load the Data File

This tutorial assumes that you have not performed any actions before following the tutorial steps. Otherwise, the names of the modules created might vary. For instructions on how to start ParaView, see section 1.2.2, Starting ParaView.

In this example, we will use a PLOT3D file. For more information on file types supported by ParaView see section 2.3, Data Types. The PLOT3D format stores the geometry (structured grid) and the attributes in separate files. First, open the geometry file by selecting File, Open from the menu bar, navigating to the folder in which the tutorial data files are located, choosing comb.xyz in the file selection dialog, and clicking Open. (The tutorial data files are available in the Download section of the ParaView web site.) You should now see the parameters page for a new PLOT3D reader.

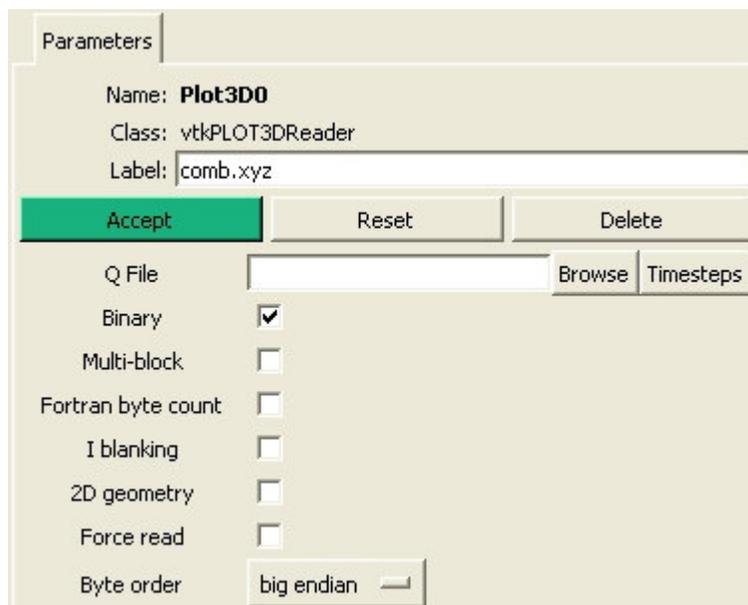


Figure 16. PLOT3D reader parameters.

So far, a few of the items on the Parameters page have been introduced as they were needed. Here is a more detailed explanation of each of the items.

- **Name:** ParaView assigns a unique name to each reader, source, and filter. This name is used by various ParaView components include the Select menu and the Selection window (when “Show source names in browsers” is selected on the Application Settings property sheet). Name can not be changed.
- **Class:** The class name of the underlying VTK object. The class name is provided for information only.
- **Label:** By default, ParaView uses this name to display modules (readers, sources, and filters) in the browser windows (Selection and Navigation windows) and in the Select menu. Although a default value is initially assigned by ParaView, the user can change the label to more easily locate modules.
- **Accept Button:** To avoid unnecessary filter execution when changing multiple parameters, ParaView does not immediately apply a filter when a parameter changes. Instead, when a change occurs, the Accept button will be highlighted in green to inform the user that one or more parameters changed. To apply these changes and execute the filter, click Accept.
- **Reset Button:** To reset the state of the module to that of the previous execution of this module, use the Reset button.
- **Delete Button:** To remove a module, use the Delete button. Note that when the output of the module is being used by another one, this button is disabled. To delete such a module, you have to delete all other filters which use its output.

The rest of the parameters are specific to the PLOT3D reader. To get more information about these, you can refer to the balloon help. If tooltips are enabled (View | Application Settings, Show tooltips), a small window describing a parameter briefly will appear after the mouse pointer is near the parameter for a few seconds. In this case, you do not have to modify any PLOT3D parameter. Click Accept. Next, change the Label to Combustor Data. (Make sure you press Enter after typing the new label; otherwise the change will not take effect.) By assigning more meaningful labels, you can make finding a module in the Selection window much easier, especially when there are many modules.

You may have noticed that all the PLOT3D parameters except the Q File are now disabled. The PLOT3D reader module behaves differently than other reader modules. When it is first created, it will allow you to set its parameters until they all match those of the file. It does this by trying to read a portion of the file with the given parameters. Until the file can be read, the Accept button remains green. This is done to inform the user that the parameters are not correct. Once the file is read, the parameters do not have to be modified anymore. Therefore, they become disabled. These steps are necessary because, unlike other file formats ParaView supports, PLOT3D files do not contain any information about the parameters used to write them.

Once the file is loaded, certain buttons in the toolbar will be enabled while others remain disabled. This is because, like the filter menu, the toolbar buttons are context sensitive.

So far, you have only loaded the geometry file. The data object that was created does not contain any data arrays (scalars, vectors, etc.). Therefore filters that require data arrays, like Array Calculator, Warp, Threshold, and Contour, are disabled. To load the attributes, click on the Browse button next to the Q File parameter, select comb.q in the file selection dialog, and click Open. Click Accept to load the attributes file. Notice that all the toolbar buttons are now enabled.

## 2. Create an Isosurface



Next, we will extract a density isosurface by using the contour filter. You can either use the contour button on the toolbar or choose Filter, Contour to create a contour filter. By default, the contour filter is created with an empty list of contour values. The parameter page should now look like this:

The image shows the 'Contour Properties' dialog box. At the top, there is a 'Scalars' dropdown menu currently set to 'Density'. Below it, the 'Scalar Range' is displayed as '0.197813 to 0.710419'. The main section is titled 'Contour Values' and contains an empty list box with 'Delete' and 'Delete All' buttons to its right. Below this is the 'Add value:' section, which includes a 'New Value' slider and text box showing '0.45412' and an 'Add' button. The 'Generate range of values:' section has a 'Number of Values' slider and text box set to '1' and a 'Generate' button. Below that is a 'Range' slider and text box showing '0.19781' and '0.71042'. At the bottom, there are three checkboxes: 'Compute Normals' (checked), 'Compute Gradients' (unchecked), and 'Compute Scalars' (unchecked).

Figure 17. Contour Properties page.

Here, the first parameter widget allows the user to choose which array to use in creating the isosurface. Any one-component array can be used to create isosurfaces. If you click on the Scalars selection widget, you can see that the available arrays are Density and StagnationEnergy. Next, the range of the currently selected array is shown. You can add isosurface values by using the Add Value widget or the Generate range of values widget. Simply move the New Value slider or type the value in the New Value entry box and click Add. To remove a value from the list, select it and click Delete. All the contour values can be removed from the list by clicking Delete All. Start by adding a density isosurface at 0.3. Move the New Value slider to 0.3 and click Add. Make sure the Compute Scalars box is checked. Next, click Accept. Also, change the label of the contour filter to Animated Contour. The ParaView window should now look like this.

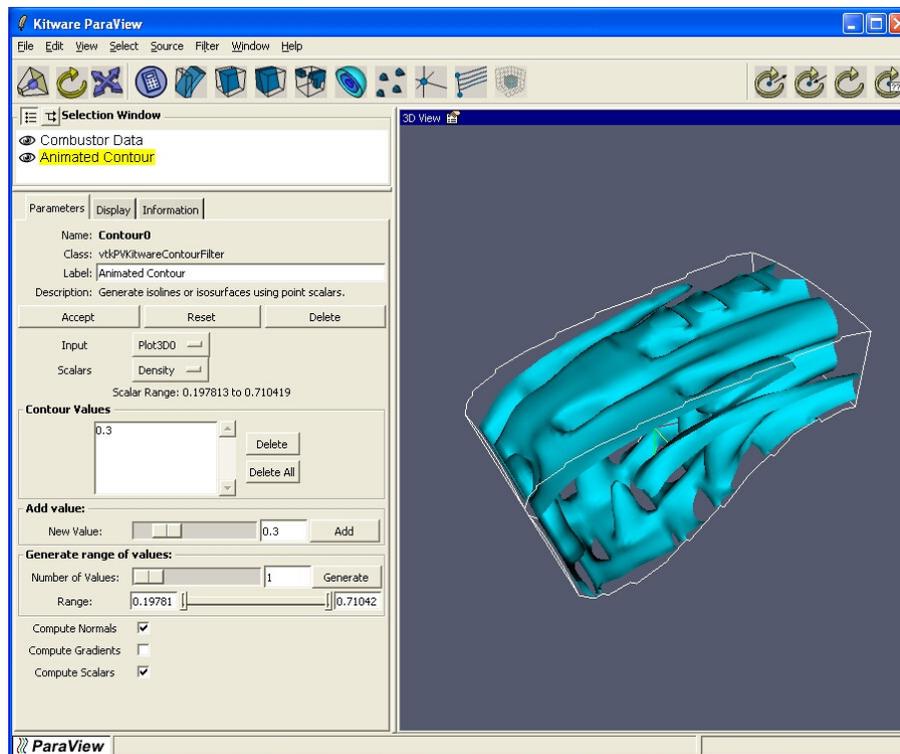
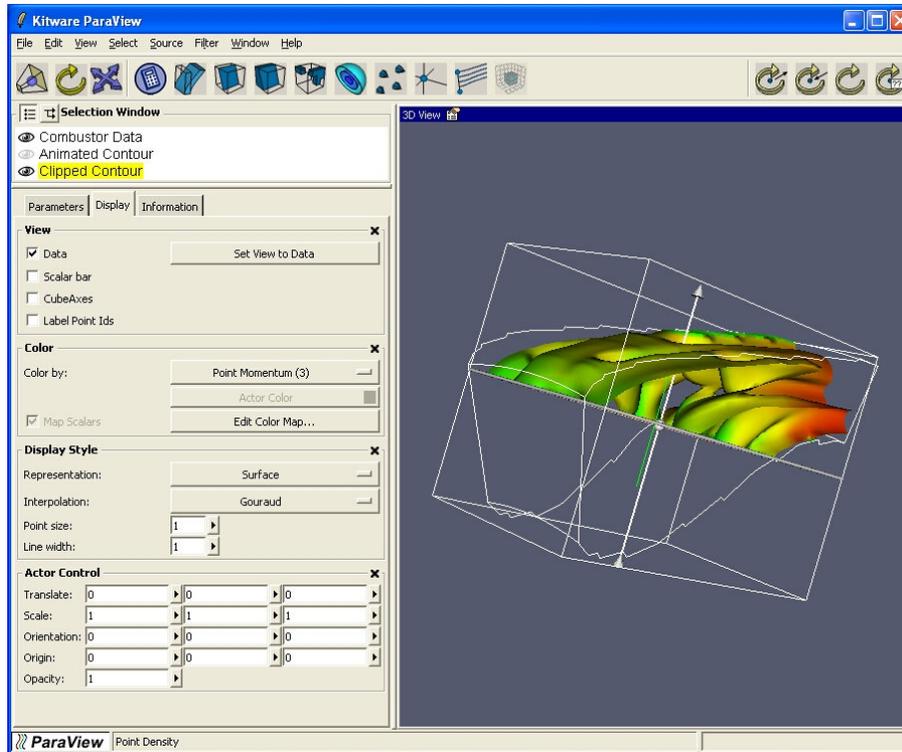


Figure 18. First contour.

### 3. Clip the Isosurface



Because the isosurface displayed covers a large portion of the whole grid, it is difficult to see some of the details at the center of the domain. You can remove part of the isosurface by using the clip filter. First make sure the contour filter (Animated Contour) is selected. Create a clip filter by either clicking on the Clip button on the toolbar or by selecting Filter, Clip. The default clipping surface is a plane. The orientation and center of the plane can be modified by either interacting with the 3D widget in the Display Area or changing the values in the appropriate entry boxes. See Appendix A: 3D Widgets for the descriptions of the most common 3D widgets. In this example, the default parameters of the plane are good. Click Accept. You will notice that the clipped iso-surface replaces the whole iso-surface. This is because when a clip filter is created, the visibility of its input is automatically turned off by ParaView. Change the label of the clip filter to Clipped Contour. Next, change the color mapping of the surface by selecting the Display tab and changing Color By to Point Momentum (3). To color by the first component of the Momentum vector, click the Edit Color Map... button, and select Component from the menu in the Vector portion of the interface. Click the Reset Range button at the top of the Color Map section to reset the color scale to the range of the X component of Momentum. (Click the Back button to return to the Display tab.) ParaView window should now look like this.

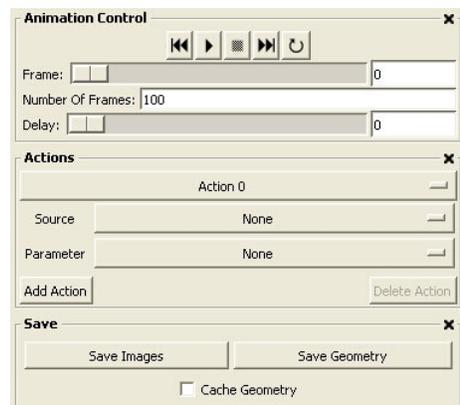


**Figure 19. Clipped contour (colored by the first component of Momentum).**

You can turn off the visibility of the plane widget by switching back to the Parameters page and disabling Visibility of the Clip Function.

#### 4. Animate the Isosurface

In this section, we will demonstrate how to create an animation that sweeps over a range of isosurface values by using ParaView's animation editor. To display the animation editor, select View, Animation. The left panel should now look like the one shown on the right.



The top set of buttons work similarly to a VCR – the first button can be used to return to the beginning, the next button will play, the center square will stop the animation if it is playing, the double right arrows will go to the end, and the final button is a toggle button which can be used to enter a looping mode. The Number of Frames entry box determines the total number of frames in the animation, and the Frame slider determines the frame currently displayed. The Delay slider controls the delay (in milliseconds) between frames.

Animations are created by adding actions to be performed when the play button is pressed. Create your first action by first selecting Animated Contour as the Source and Contour Values as the Parameter. Next set the Start Value to 0.3 and End Value to 0.6. Here is what the action you just created does: set the Contour Values parameter

(currently, only one iso-surface value can be changed from the animation editor) of the Animated Contour filter to 0.3 when the first frame is displayed and to 0.6 when the last frame is displayed; for any frame in between, linearly interpolate the iso-surface value between 0.3 and 0.6. This is a very simple key-frame animation model in which the first and the last frames are used as the key-frames and only linear interpolation is allowed. You can add as many items as you want by using the Add Action button and describing the action using the Source, Parameter, Start Value, and End Value parameters. Next, reduce the number of frames to 20.

To play the animation, press the play button on the VCR control panel. You will notice that the animation plays relative slowly. This is because, at each frame, a new isosurface is created by running the contour filter. This is a relatively expensive operation. It is possible to avoid this by caching the polygonal data representing each isosurface when the contour filter is run the first time. Make sure that caching is on by checking Cache Geometry at the bottom of the animation editor page. (It is on by default.) The first time you ran the animation, it was be slow, but the isosurfaces were cached because Cache Geometry was enabled. The next time you run the animation, you will notice that it runs much faster. The drawback of using geometry caching is in that the cache stores the geometry of every step in memory. When the number of steps or the geometry is large, ParaView might run out of memory. To remove the geometry cache, disable the Cache Geometry option.

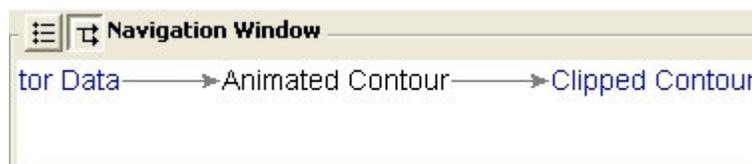
## 5. Delete the Contour and Clip Filters

Before moving to another animation, we will demonstrate how to delete existing ParaView modules. First, display the Source page on the left panel by selecting View, Source. Next select the Animated Contour object in the Selection window and select the Parameters tab. You will notice that the Delete button is disabled.



**Figure 20.** If the output of a source is used by another filter, that source can not be deleted.

This is because the output of the contour filter is used by the clip filter. You can verify this by switching to the Navigation window.



**Figure 21.** Navigation window showing part of the pipeline.

ParaView does not allow the deletion of a module if its output is used by another module. To delete the contour filter, first select the clip filter (Clipped Contour), delete it, and then delete the contour filter. It is also possible to delete all modules by selecting Edit, Delete All Modules. However, since we will need the reader module in this example, do not use Delete All Modules.

## 6. Animate a Cut-Plane



In this section, you will create a cut-plane through the combustor dataset and move it in an animation. Create a cut filter by either clicking on the Cut button on the toolbar or selecting Filter, Cut. Change the label to Cut-plane, turn off the visibility of the 3D widget by clicking on Visibility in the Cut Function property, and accept all other default parameters by clicking Accept. The ParaView window should look like this.

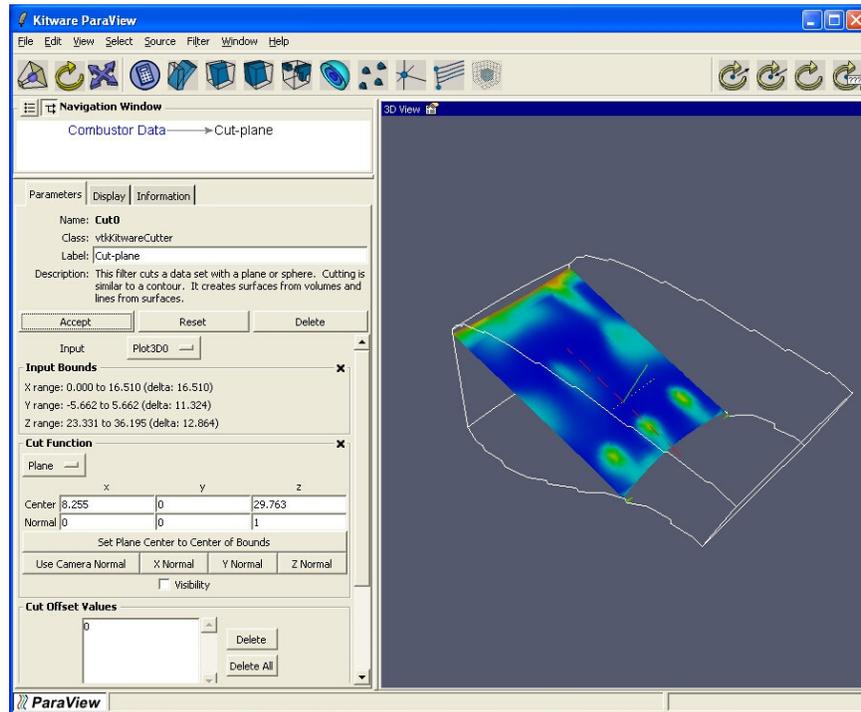


Figure 22. Cut-plane to be animated.

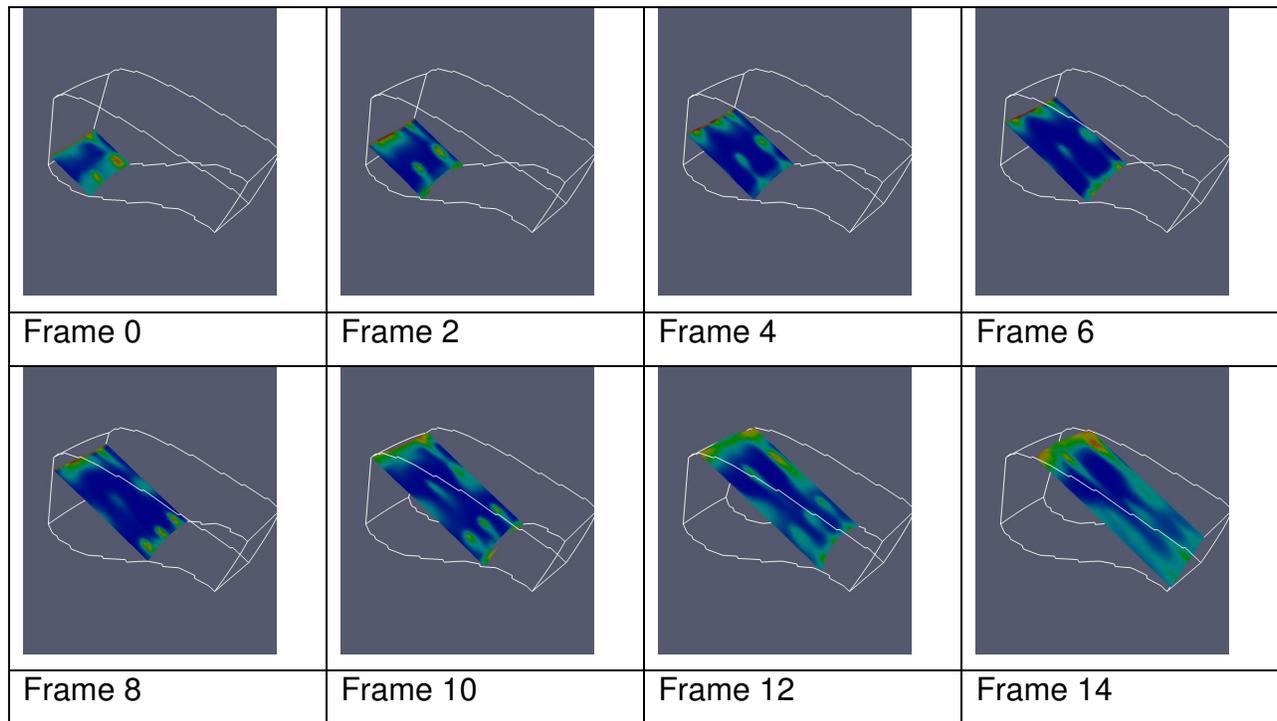
Next, switch to the animation editor by selecting View, Animation. You will notice that the previous action is deleted. Anytime a module is deleted, ParaView looks for actions that use that module. All matching actions are deleted. Add a new action by selecting Cut-plane as the source, Cut Values as the parameter, -5 as the Start Value, and 5 as the End Value. Make sure that Cache Geometry is turned on and press play in the VCR controls. Next time you play the animation, it should run much faster. In fact, the playback speed might be too high on a fast computer.

To add a delay between each frame, you can use the Delay slider. Move the Delay slider to 100 to add a 100 ms pause between each frame. Next time you play the animation, you will notice that it will run slower.

It is also possible to further control what happens during each frame by using the Script Editor. It allows you to write a Tcl script that will be executed at every frame. It is mainly for advanced users, and will not be discussed in further detail here.

ParaView can save each frame as an image. These images can later be collected to create an animation file such as AVI or MPEG using commonly available utilities. To write out such images, first prepare the animation; next, make sure that the Display

Area is not covered by another window and finally click on Save Images. ParaView will run the animation and save an image file for each frame. Some of the frames from this section's exercise are displayed below.



**Figure 23. 8 frames from the animation.**

This concludes the third tutorial. You can now exit ParaView by either selecting File, Exit or closing the window.