ParaView Documentation

Table Of Contents

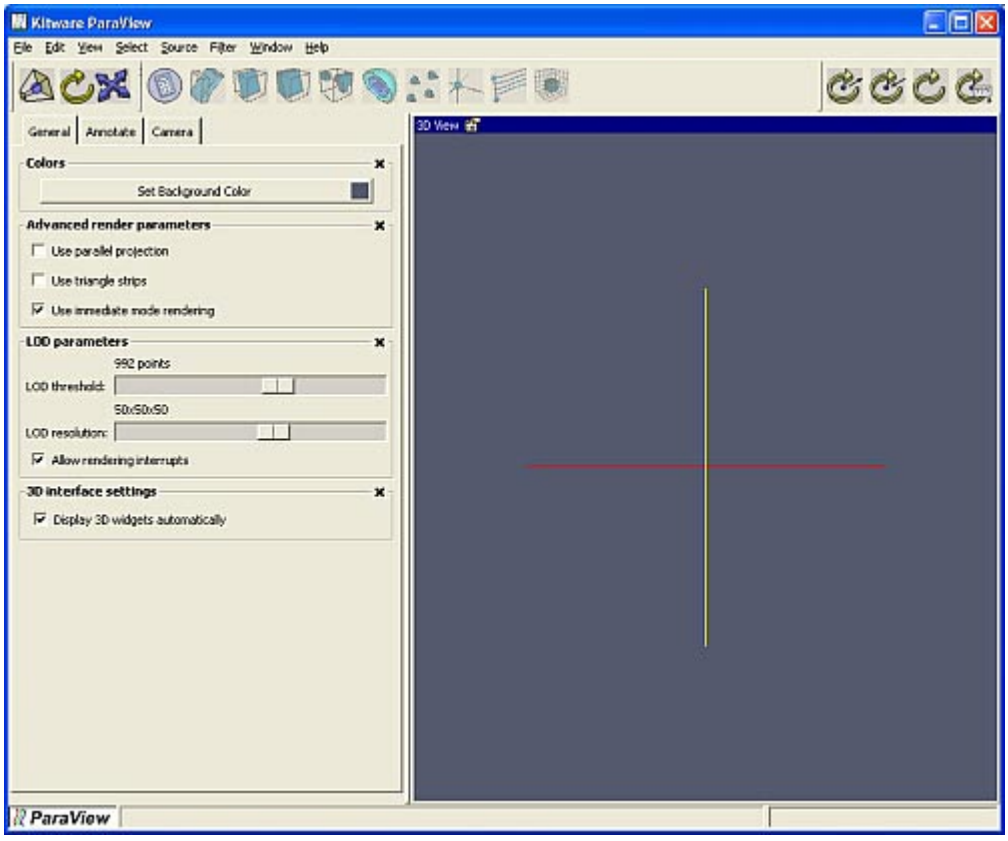
User's Guide

Introduction

System Overview

ParaView is a turnkey application developed by Kitware that is built on top of the Visualization Toolkit (VTK). ParaView is designed to handle large data sets by distributing the data across multiple computers, but can also be used as a stand-alone single-process application.

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 (i.e. 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.



At startup, ParaView will appear as shown above. There are several regions to 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.

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 button is described in more detail below.

File: The File menu can be used to load and save data files, export VTK scripts, 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 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 three 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 cancel this operation.

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 available in the Filters

menu will change to correspond with 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 cancel this operation.

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 this Help or version information on ParaView.

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.

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 contents 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.

The Display Area

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

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.

A Simple Example

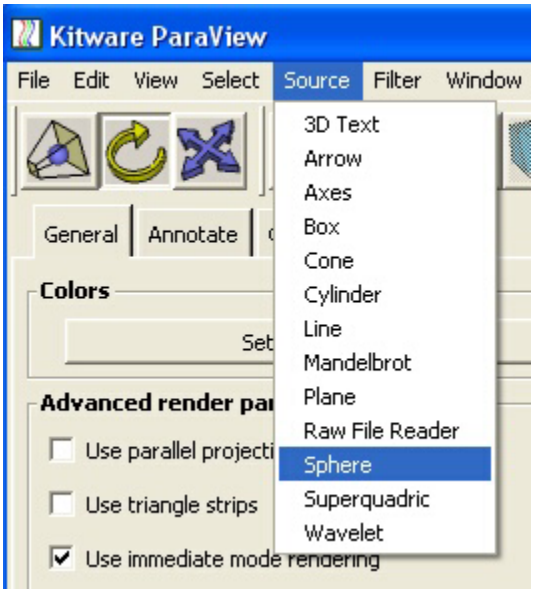
A simple ParaView example will be presented in this section in order to familiarize you with the basic operation of the application. More detailed information can be obtained by using the hyperlinks in this section. A separate tutorial contains several detailed examples which should be attempted after you are comfortable with the basics.

Step 1: Start ParaView

If you have installed ParaView on Windows, you can start the application by using the Start menu. Under the Programs submenu you will find a ParaView06 menu which contains ParaView and Check for updates. Select the ParaView item and you will see the ParaView splash screen. After a few seconds, the application should appear.

If you have installed ParaView on Linux or Unix you will need to go to the directory in which the application is installed, and type ./ParaView.

Step 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. The Accept button on this property sheet will be highlighted in red, 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

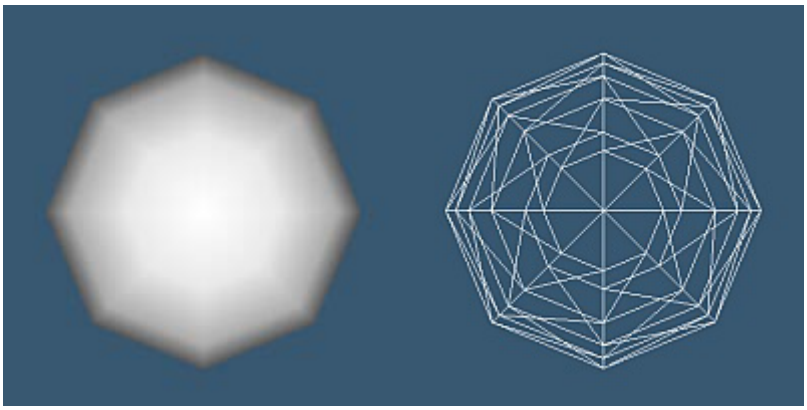
For users who are familiar with VTK, here are the details of what has happened. A vtkSphereSource was created, and the parameters of this object are presented to you in the property sheet. When you first create the vtkSphereSource it has an empty output, and therefore you need to press the Accept button 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.

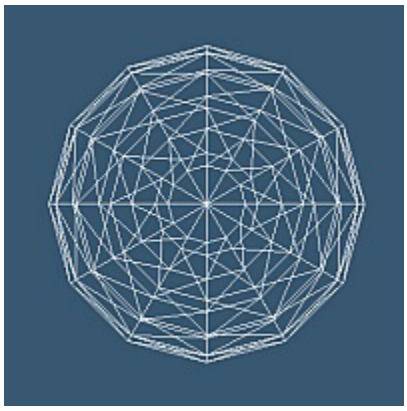
When you add data to ParaView (by loading, creating a source, or filtering) a set of objects are created to manage the data generation, data storage, and display. ParaView groups these objects together under the name presented in the property sheet - in this simple example it is Sphere1. When you select 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 in 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.

Step 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, press the Display tab on the property sheet for the sphere shown in the Left Panel. In the Display Style area, change the Representation from Surface to Wireframe. The image should now appear similar to the one shown below on the right.



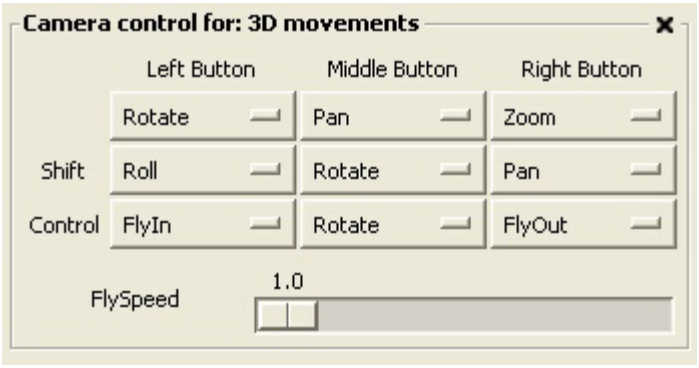
Step 4: Change the resolution



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, both of which have 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 red 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 to 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 red, 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).

Step 5: Interact with the sphere

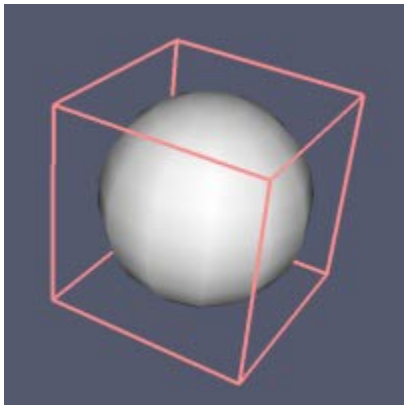


Using various mouse and keyboard combinations you can control the objects and the camera in the Display Area. The mouse bindings can be viewed and changed by selecting the 3D View Properties option from the View menu, then clicking on the Camera tab. You will see that there are two "styles" of mouse bindings: one for 2D movements and one for 3D movements. By default the camera is in 3D movement mode, and the lower 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.

Step 6: Add a bounding box



In this step we are going to add a filter to the pipeline. Select the Outline item from the Filters 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. The Outline0 object is the active data object (if another filter is added it will be connected to this data object) and is highlighted in yellow.

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 slider 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 you can either click on Sphere2 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.

Step 7: Shrink the sphere

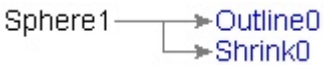
We will now apply another filter to the sphere. First, make sure that Sphere1 is the highlighted item in the Selection Window so that the filter we

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, and 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.

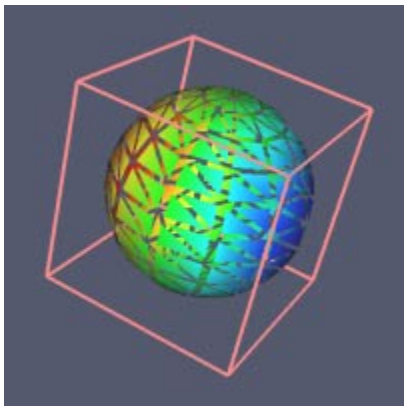
In this example you have created a branching pipeline in ParaView. You can see this in the Navigation Window. To change from Selection Window to Navigation Window, click on the  icon next to the Selection Window title. You should see the title change to Navigation Window, and in the window you should see:

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:



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

Step 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 1. 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 Y value of the Normal to look up a color between blue and red.

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.

Command Line Arguments

There are several command line arguments that can be used to control the execution of ParaView. These are group 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.

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.

--help : Displays available command line arguments.

Parallel

--use-rendering-group, **-p** : Use a subset of processes to render. This is useful on systems/clusters which have a limited number of rendering nodes. It allows to run ParaView on more nodes than available rendering nodes while only rendering on the subset.

--group-file, **-gf** : When using the --use-rendering-group options, the number of nodes to render with is read from this file (usage --group-file=fname).

--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.

Mesa

--use-software-rendering, **-r** : Use software (Mesa) rendering on all nodes. This is useful when the user wants to have a window only on the first node's display. This is accomplished with using this option in combination with the **PV_OFFSCREEN** environment variable.

--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 with using this option in combination with the **PV_OFFSCREEN** environment variable. Furthermore, since the first node uses hardware accelerated rendering, the performance is not compromised when rendering locally.

Running ParaView With MPI

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 with 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 FAQ on the ParaView web page for up-to-date information on how to enable this feature.

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. However, unless the DISPLAY variable is :0 or localhost:0, all windows will appear on the same display and ParaView will not function properly. 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 solution to these problems is:

1. to 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-),
2. on each process, set a DISPLAY variable which depends on where the process is running and if necessary, set-up authorization in a a start-up file which belongs to the user which starts the MPI job (for example .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. See the FAQ on the ParaView web page for up-to-date information.

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 of starting processes and sometimes recompiling the service (obviously not an option if no source code is available).

Environment Variables

ParaView looks for a few environment variables that can be used to modify its behavior. Environment variables are only accessed from the first process which

contains the user interface. ParaView is responsible for broadcasting the effects of the modifications to all processes.

PV_OFFSCREEN

This environment variable is important only when running with multiple processes. It determines whether off screen rendering will be used for the satellite processes. The default behavior is for each node to map a window and use hardware rendering. When the assigned processors do not have access to a display, this environment variable should be set to "True" (any value will do). When this variable is set, ParaView only maps the render window of the first process. Software render will be used for the other processes. Of course, using software rendering will affect ParaView's rendering performance.

It is important to note that currently, you must link against Mesa's OpenGL libraries to get off-screen rendering on Unix systems. See ParaView FAQ on the web page for more up-to-date information.

PV_SEPARATE_RENDER_WINDOW

Setting this environment variable to "True" (or any value) will cause ParaView to create a render window separate from its main interface window.

Check For Updates

For ParaView users on Windows systems, it is easy to update to the latest released version. On the Start menu under ParaView06 you will find a **Check for Updates** menu item. After selecting the Check for Updates option, a dialog box will appear informing you that you must have an active internet connection. Click on the Next button to proceed. At this point you will either see a dialog box indicating that you are running the latest available version of ParaView (no updates are available) or a dialog box containing the release notes for the version you are about to download. After reading the release notes you can decide to proceed with the download or cancel it.

Once the download is complete, the Install Shield Wizard will automatically start, leading you through the process of installing the new version.

Linux and Unix users can stay up-to-date with new releases of ParaView by visiting the ParaView web site.

ParaView Support

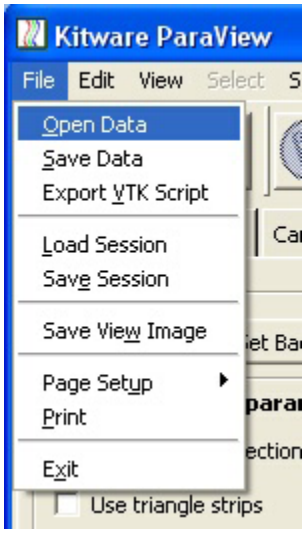
A searchable list of Frequently Asked Questions (FAQ) and the corresponding answers is available here: http://public.kitware.com/cgi-bin/paraviewfaq. This is the first place you should look if you have a question or encounter a problem.

The ParaView mailing list is the principal means of communication among users and developers. Please go to http://public.kitware.com/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 518-373-4194 or kitware@kitware.com for more information.

Loading Data

Overview



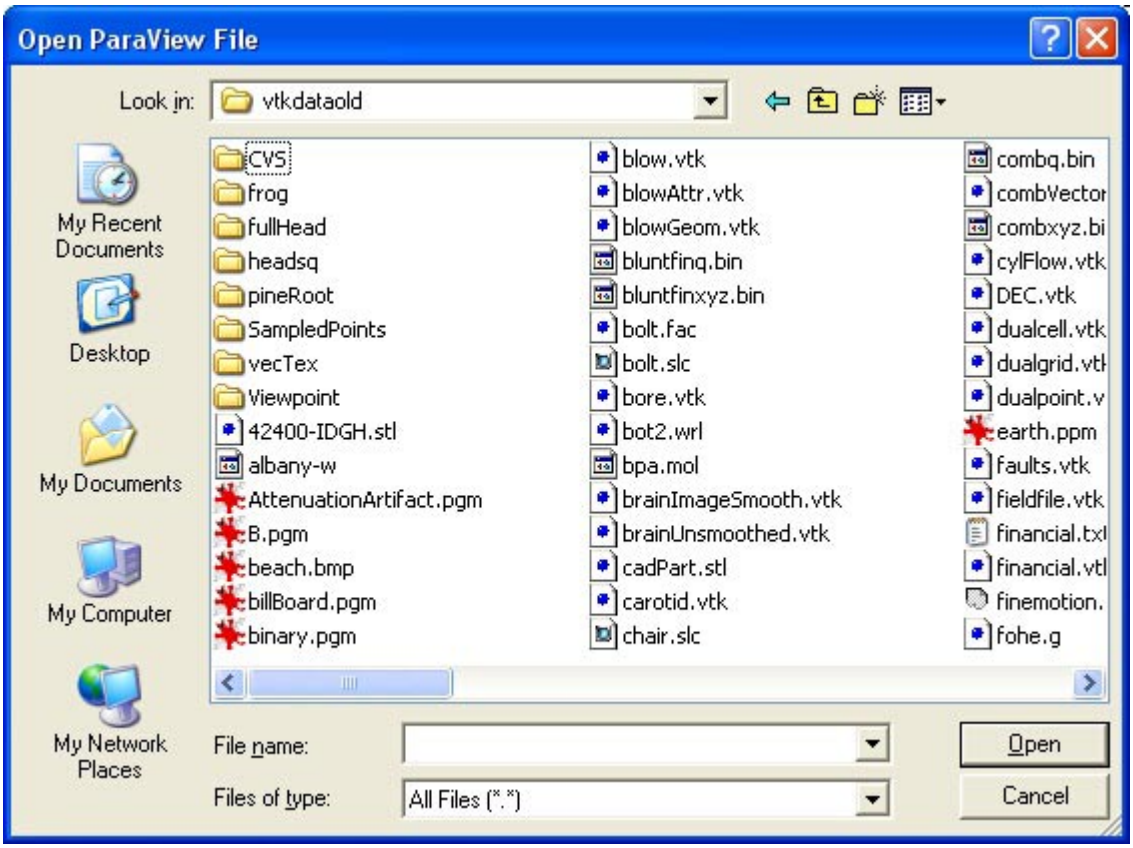
There are several ways to load data into ParaView. One method is to use the Open Data option from the File menu on the Menu Bar to specify a data file. Details on this method including the supported data types are provided in the next section.

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.

Raw regular rectilinear grid data can be read into ParaView using the Raw File Reader source, which is covered in the next chapter.

When either the Open Data or Load Session options are selected, a OS-specific dialog box with 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 can be used to cancel the load operation.



Supported Data Formats

Using the Open Data option from the File menu will cause the Open ParaView File dialog box to pop up. Using this dialog you can select a data file to load into ParaView. The supported data types are listed here.

Parallel VTK Files: This is the "old" (VTK 4.0) data format for parallel VTK files. All types of VTK data (PolyData, ImageData, etc.) are stored with the same file extension. Files of this type will have a *.pvtk extension. The data object created by this reader may have a type of vtkPolyData, vtkImageData, vtkUnstructuredGrid, vtkStructuredGrid, or vtkRectilinearGrid.

VTK Files: This is the "old" (VTK 4.0) data format for VTK files. All types of VTK data are stored with the same file extension. In the "new" format, each data type has a unique extension. Files of this type will have a *.vtk extension. The data object created by this reader may have a type of vtkPolyData, vtkImageData, vtkUnstructuredGrid, vtkStructuredGrid, or vtkRectilinearGrid.

Stereo Lithography: ParaView can read binary or ASCII Stereo Lithography files. These files will have a *.stl extension. The data object created by this reader will be a vtkPolyData.

BYU Files: ParaView can read MOVIE.BYU files. These files will have a *.g extension, and the data object created by the reader will be a vtkPolyData.

POP Ocean Files:

EnSight Files:

PLOT3D Files: ParaView can read PLOT3D files. PLOT3D is a computer graphics program designed to visualize the grids and solutions of computational fluid dynamics. These files will have a *.xyz file extension, and the output data object will be a vtkStructuredGrid.

VTK PolyData Files: This is the "new" file format for polygonal data in VTK. The file extension will be *.vtp, and the output data object will be a vtkPolyData.

VTK Unstructured Grid Files: This is the "new" file format for unstructured grid data in VTK. The file extension will be *.vtu, and the output data object will be a vtkUnstructuredGrid.

VTK ImageData Files: This is the "new" file format for regular rectilinear grid data (images and volumes) in VTK. The file extension will be *.vti, and the output data object will be a vtkImageData.

VTK StructuredGrid Files: This is the "new" file format for structured grid data in VTK. The file extension will be *.vts, and the output data object will be a vtkStructuredGrid.

VTK Rectilinear Grid Files: This is the "new" file format for rectilinear grid data in VTK. The file extension will be *.vtr, and the output data object will be a vtkRectilinearGrid.

PVTK PolyData Files: This is the "new" file format for polygonal data in VTK. The file extension will be *.pvtp, and the output data object will be a vtkPolyData.

PVTK Unstructured Grid Files: This is the "new" file format for parallel unstructured grid data in VTK. The file extension will be *.pvtu, and the output data object will be a vtkUnstructuredGrid.

PVTK ImageData Files: This is the "new" file format for parallel regular rectilinear grid data (images and volumes) in VTK. The file extension will be *.pvti, and the output data object will be a vtkImageData.

PVTK StructuredGrid Files: This is the "new" file format for parallel structured grid data in VTK. The file extension will be *.pvts, and the output data object will be a vtkStructuredGrid.

PVTK Rectilinear Grid Files: This is the "new" file format for parallel rectilinear grid data in VTK. The file extension will be *.pvtr, and the output data object will be a vtkRectilinearGrid.

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.

Session files contain the sequence of steps that were performed in ParaView to read the current state. This includes all creation and deletion of objects, parameter changes, etc. This is not the most efficient way to store the current state of ParaView, but in this release it is the only method available.

Sources

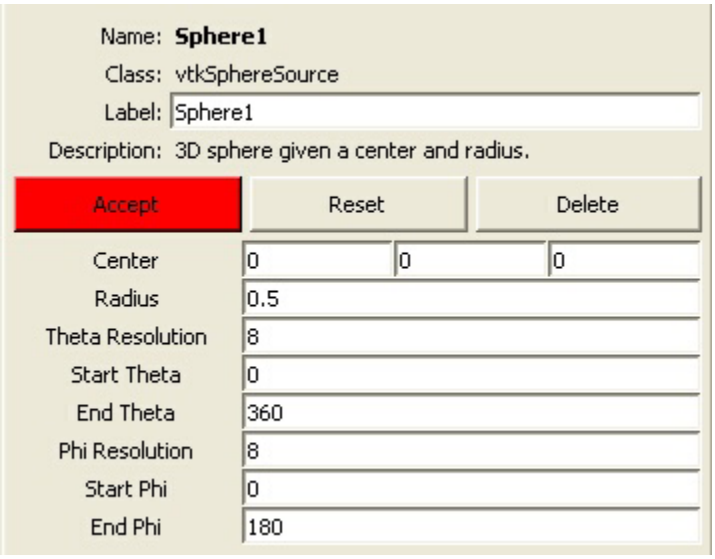
Overview

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:

1. The property sheet for the source will be displayed. On this property sheet you will see the configurable parameters of the source.
2. The Selection / Navigation Window will be displayed if it was not yet displayed already.
3. 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 red whenever a source is first created or a parameter is changed that would require updating. 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, and is the name of the underlying Tcl object created. The class indicates the type of VTK object. The label by default will match the name of the object, but this can be changed by typing a new label in the text entry box. This label will be used when displaying lists of data objects in the Select menu and the Selection / Navigation Window. 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 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 in pressed. Note that the Accept button will turn red 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 this indicates that a filter relies on the output of this source and therefore this source cannot be deleted until after the filter is deleted.

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

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

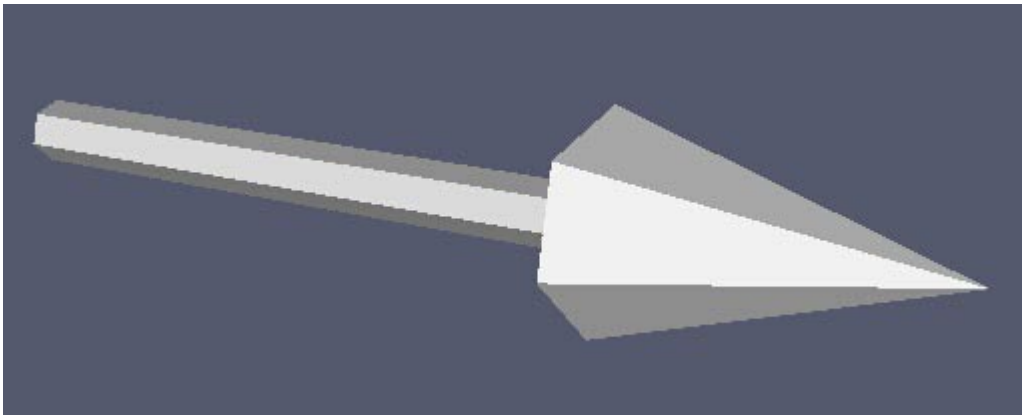
operation of the camera. The object created by this source is a **vtkVectorText** object, and the output of this source is **vtkPolyData**. An example of 3D Text is given below.

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.

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.

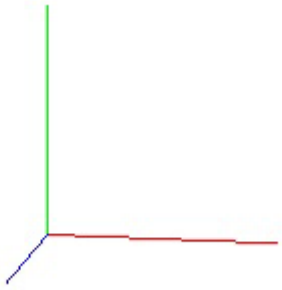
Shaft Resolution: This is the number of faces around the cylinder representing the shaft of the arrow. The default value is 6.

Shaft Radius: The 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.

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 red line, the Y axis as a green line, and the Z axis as a blue 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:

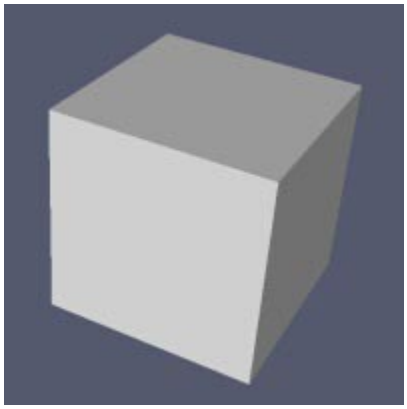
Scale: By default the axes lines have a length of 1, or if the symmetric option is selected, a length of 1 in each direction for a total length of 2. This value can be changed to make the axes bigger or smaller.

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 both 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.

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 used 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 specified the length of the box along the X axis. By default the X length of the box is 1.

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

Z Length: This value specified 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 on (0,0,0).

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 used 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:

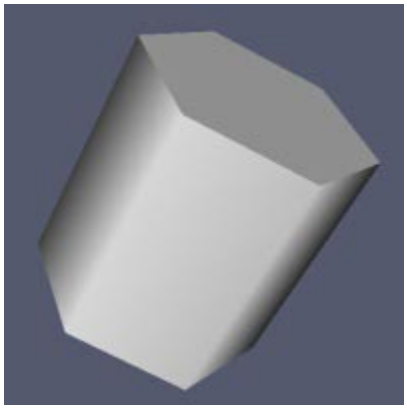
Resolution: This value indicates the number of divisions around the cone. The higher this number, the closer the polygonal approximate will come to representing a cone, and the more polygons it will contain. The default resolution is 6.

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 (there is an N sided polygon closing off the cone where N is the resolution) or open. By default the cone will be capped.

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 used 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.

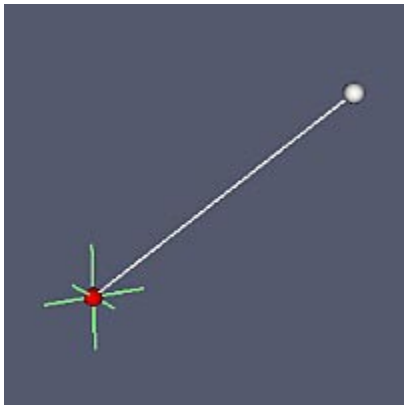
Radius:This is the radius of the cylinder. The default value is 0.5.

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 (there are two N sided polygons closing off the cylinder where N is the resolution) or open. By default the cylinder will be capped.

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 used 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 defined 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 simple 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:

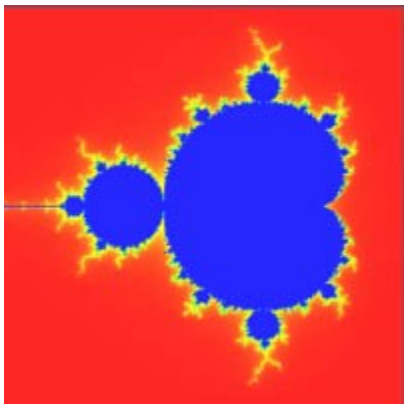
> **Point1**: These three values indicate the coordinates of one of the two end points of the line.
>
> **Point2**: These three values indicate the other end point coordinate of the line.
>
> **Visibility**: This check box can be used to toggle the visibility of the interaction widget. When the interaction 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 Ball**: 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 left 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.

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 (represented by 2) is mapped to the Z axis. The imaginary component of X is represented by 3. All values entered must be between 0 and 3 inclusive.

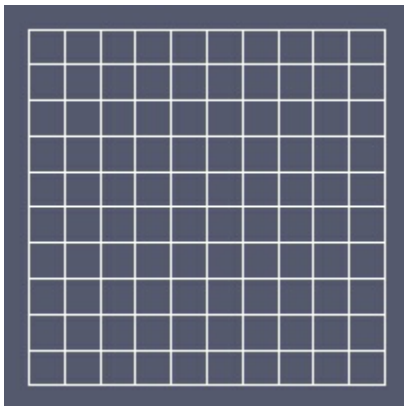
Origin: The four values indicate the values of C (real and imaging) and the initial value X (real and imaging). The first two number represent the real

and imaging components of C, with the last two indicating the real and imaging 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.

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 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 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.

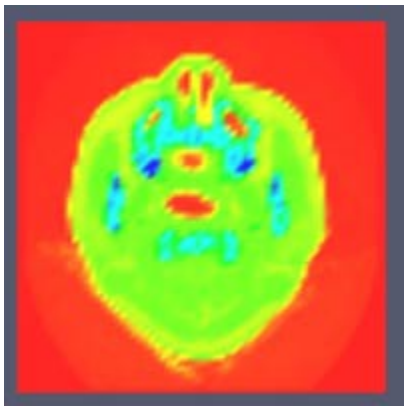
Second Point:This coordinate represents the location of a third corner of the parallelogram. One edge of the parallelogram will be the line connecting the origin with this second point, and this will be considered the Y axis of the parallelogram.

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.

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

Raw File Reader



The Raw File Read can be used to read raw 2D or 3D data defined on a regular rectilinear grid. The data can be in a single image file or a volume can be defined with multiple image files. Although this is technically a reader and should therefore be handle by Open Data, many parameters other than the file name are necessary for reading raw data. Grouping this reader with the other source object allows for a property sheet on which parameters can be specified. A file prefix and pattern are used to determine the names of the files in the series. The origin, extent, and spacing of the data must be specified along with the number of components. If the image file is bigger than expected from the extent and number of components, the extra bytes will be considered a header at the beginning of the file and will be skipped.

If a 2D image is specified using the extent, then the image will be mapped onto a plane and displayed. If a 3D image is specified, the bounding box of this volume will be displayed. The image plane 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 **vtkImageReader** object, and the output of this reader is **vtkImageData**. . An example of a 2D image read using this raw file reader is given above.

The Raw File Reader has the following parameters:

File Prefix: Type the path and base file name of the file(s) you wish to read. You can also use the Browse button to locate a file. Be sure to remove any extension that will be added by the file prefix. For example, if you have a set of files image.1, image.2, image.3, etc located in /home/data, the File Prefix would be /home/data/image and the pattern would supply the .1, .2, .3, etc.

File Pattern: This is the pattern to use when creating the file names. By default this is set to %s.%d indicating that a "." and a number will be

appended to the file name given by the File Prefix. The number will range across the Z extent of the data. If you specify the exact file name for a 2D data file in the File Prefix, you can change the File Pattern to %s to read that single file.

Data Type: The Data Type pulldown allows you to select from the following supported data types: char, unsigned char, short, unsigned short, int, unsigned int, long, unsigned long, float, and double. This must be specified accurately for the read to succeed.

Byte Order: The byte ordering my be Little Endian or Big Endian. If this is specified incorrectly, the will likely appear quite "noisy".

Origin: The coordinate location of the origin of the image or volume can by specified here. The default value is (0,0,0) and it is fine to leave this default if you do not know the origin of your data.

Spacing: The spacing values indicate the world coordinate distance between sample points in the X, Y, and Z directions. The default values are all 1. If you do not specify the spacing correctly, the image you see may appear stretched or shrunk along an axis direction.

Extent: The six extent values indicate the minimum and maximum X index values, the minimum and maximum Y index values, and the minimum and maximum Z index values. If these values are X1, X2, Y1, Y2, Z1, Z2, then each image file is considered to be (X2-X1+1) by (Y2-Y1+1) elements (with X changing fastest) and there are (Z2-Z1+1) images with the %d value in the file pattern ranging from Z1 to Z2 (inclusive). The extent must be correctly specified for the read succeed, except that the Z extent can be reduced. For example, if you have a set of 100 images with names image.1, image.2, image.3, etc. you can specify a Z extent of 50, 50 to read in only the 50th image in the series.

Num Components: This entry box indicates the number of components per scalar value in the raw data. By default this value is 1. If you are loading I (intensity) data this value is 1, if you are loading IA (intensity alpha) data the number of components is 2. RGB data has 3 components, and if you are loading RGBA data this value would be 4.

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 starting / ending angles can be changed used 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 Point Normals 1.

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.

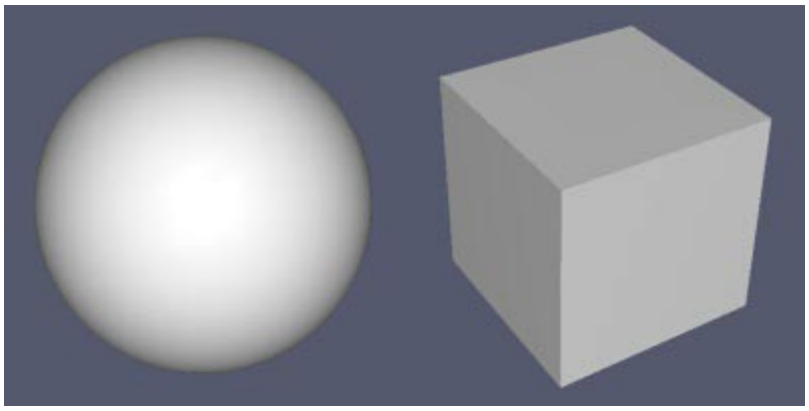
Phi Resolution: This number represents the number of division between start phi and end phi on the sphere. The phi divisions are similar to latitude lines on the earth. The default phi resolution is 8.

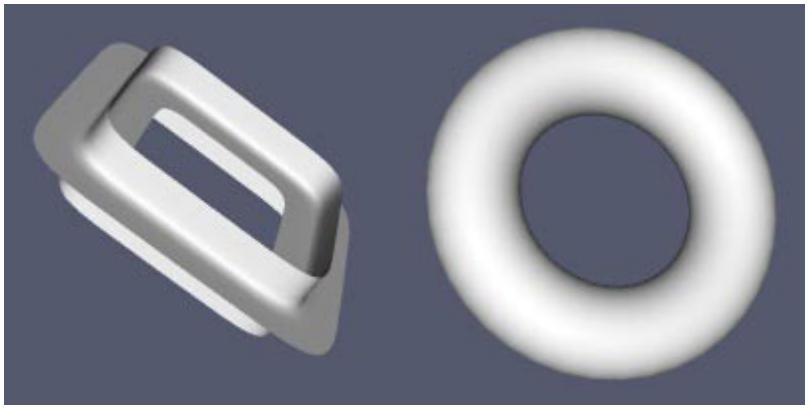
Start Phi: To form a complete sphere the start phi should be 0 and the end phi should be 180. The start phi can be adjusted to form only a portion of a sphere. The default value is 0.

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.

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.

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.

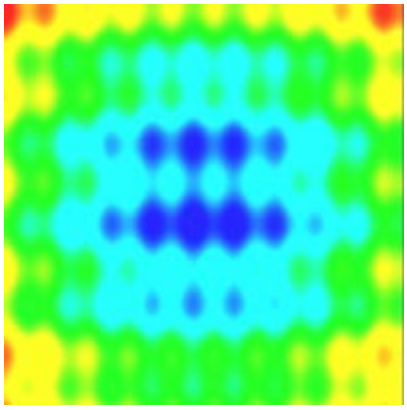
Theta Roundness: Define the roundness in the theta direction. A value of 0 represents a rectangular shape, a value of 1 represents a circular shape, and higher values represent higher order shapes. The default value is 1.

Phi 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 higher values represent 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 do 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.

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:

Dimensions: This is the extent of the output. 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.

Filters

Overview

The Filters menu in the ParaView Menu Bar contains a list of filters that can be applied to the active data object. In addition, a set of filters are available from the toolbar. The list of filters available in the Filters 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 Filters menu or press a filter button in the Toolbar, several things will occur:

1. The property sheet for the filter will be displayed. On this property sheet you will see the configurable parameters of the filter.
2. The Selection / Navigation Window will be displayed if it was not yet displayed already.
3. 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 red 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.

In this version of the documentation only a very brief description is provided for each filter. This will be expanded in the next release of ParaView.

Cell Centers

The Cell Centers filter takes as input any data set and generates on output points at the center of the cells 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 will be associated with the points on output.

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 using a particular point. Optionally, the input cell data can be passed through to the output as well.

Clean

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

Connectivitiy

The Connectivity filter extracts cells that share common points and/or meet other connectivity criterion. (Cells that share vertices and meet other connectivity criterion such as scalar range are known as a region.) The filter works in one of six ways: 1) extract the largest connected region in the data set; 2) extract specified region numbers; 3) extract all regions sharing specified point ids; 4) extract all regions sharing specified cell ids; 5) extract the region closest to the specified point; or 6) extract all regions (used to color the data by region)

Crop

This filter will crop a volume of interest out of a larger structured data set.

Decimate

The Decimate 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 vtkDecimatePro is a vtkPolyData object, and only triangles are treated. If you desire to decimate polygonal meshes, first triangulate the polygons with vtkTriangleFilter object.

The implementation of vtkDecimatePro is similar to the algorithm originally described in "Decimation of Triangle Meshes", Proc Siggraph `92, except that this algorithm does not necessarily preserve the topology of the mesh and it is guaranteed to give the a 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., deleted followed by hole triangulation using edge collapse). This continues until the priority queue is empty. Next, all remaining vertices are processed, and the mesh is split into separate pieces along sharp edges or at non-manifold attachment points and 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; 2) mesh splitting is enabled; 3) the algorithm is allowed to modify the boundary of the mesh; and 4) the maximum allowable error is set to VTK_LARGE_FLOAT. Other important parameters to adjust include the Feature Angle and Split Angle, since these can impact the quality of the final mesh. Also, you can set the Accumulate Error to force incremental error update and distribution to surrounding vertices as each vertex is deleted. The accumulated error is a conservative global error bounds and decimation error, but requires additional memory and time to compute.

Elevation

The Elevation filter can be used to generate scalar values from 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

Extract Edges

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

Extract Surface

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

Extract VOI

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 less than

one will super sample the original data, while sampling rates greater than one will sub sample the data. The Extract VOI filter can be accessed through the Toolbar or the Filters menu.

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

Generate Normals

The Generate Normals 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.

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 extrude in the direction of a user specified

vector, towards a point, or in the direction of vertex normals . 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, 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.

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 NumberOfSubdivisions. 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 filter approximates point data using the same scheme. New triangles create at a subdivision step will have the cell data of their parent cell.

Mask Points

The Mask Points filter is a filter that passes through points and point attributes from 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.

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.

Outline Corners

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

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.

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.

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.

Reflection

The Reflection filter reflects a data set across one of the planes formed by the data set's bounding box. Since it converts data sets into unstructured grids, it is not efficient for structured data sets.

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.

Rotational Extrusion

The Rotational Extrusion filter is a modeling filter. It takes polygonal data as input and generates polygonal data on 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.

Shrink

The Shrink filter shrinks cells composing an arbitrary data set towards their 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 general data set type vtkUnstructuredGrid.

Shrink (Polygons)

The Shrink (Polygons) filter shrinks cells composing a polygonal data set (e.g., vertices, lines, polygons, and triangle strips) towards their 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 data set type of this filter is polygonal data.

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.

Triangle Strips

The Triangle Strips filter generates triangle strips and/or poly-lines from input polygons, triangle strips, and lines. Input polygons are assembled into triangle strips only if they are triangles; other types of polygons are passed through to the output and 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.

Subdivide

The Subdivision filter generates output by subdividing its input polydata. Each subdivision iteration create 4 new triangles for each triangle in the polydata.

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; or if a three-dimensional data set, all cells in the 3D data set are convex with planar facets

Triangulate

The Triangle filter generates triangles from input polygons and triangle strips. The filter also will pass through vertices and lines, if requested.

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. You can also specify which of the sides are visible. This is useful for generating interesting striping effects. Other options include the ability to cap the tube and generate texture coordinates. Texture coordinates can be used with an associated texture map to create interesting effects such as marking the tube with stripes corresponding to length or time.

Warp Scalar

vtkWarpScalar is a filter that modifies point coordinates by moving points along point normals by the scalar amount times the scale factor. Useful for creating carpet or x-y-z plots.

If normals are not present in 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 UseNormal boolean to true.

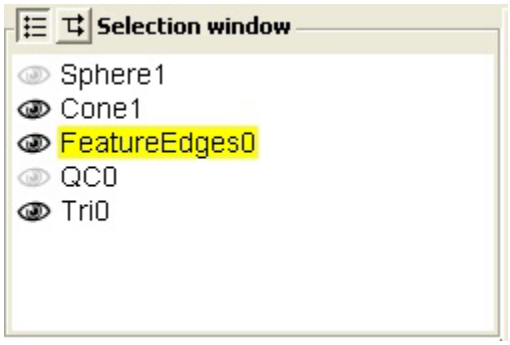
If XYPlane boolean is set true, then the z-value is considered to be a scalar value (still scaled by scale factor), and the displacement is along the z-axis. If scalars are also present, these are copied through and can be used to color the surface.

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.

Selection And Navigation

Overview

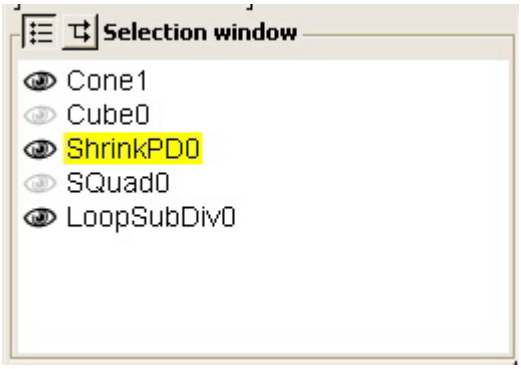
The Selection / Navigation Window will be displayed on the top portion of the Left Panel when there is data loaded or created in ParaView, the Left Panel is not hidden, and a Source or Filter property sheet is currently displayed in the Left Panel. The Selection / Navigation Window will appear similar 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 would 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.

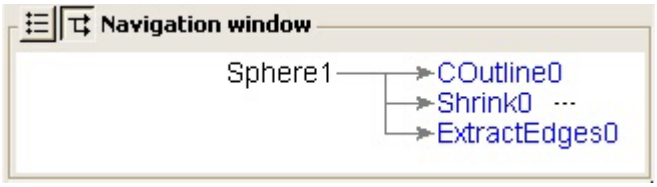
Selection Window



The first icon button in the Selection / Navigation Window can be 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.

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 (for example the reader, the source, or the filter), the data object created by that object, 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 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 such as position, orientation, color, representation style, shading type, and color map options on the Display tab.

Navigation Window



The second icon button in the Selection / Navigation Window can be 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, we have a sphere source to which three filters were applied. The Sphere1 item is shown in black which indicates 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 get to any item 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 ExtractEdges1 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 ExtractEdges1, then toggle the mode back to Navigation Window.

Toolbar

Overview

Directly below the main Menu Bar of ParaView you will find the Toolbar. On the Toolbar are some of the 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 Preferences tab of the Application Settings property sheet. In the below example these settings indicate a flat frame and flat buttons.

Based on the current state of ParaView (is there a current data object, and if so what type is it?) 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 which is unstructured.

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.

Camera Controls

The first three buttons in the toolbar, as shown on the right, are the camera controls. The first item is a button which can be pressed to reset the view. The second two can be used to switch between a 3D and a 2D interaction mode. 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.


The 3D movements 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 movements 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.

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 available at all times. Filters that cannot be applied because the active data object is of an incompatible type will have the corresponding shortcut button disabled, and will appear grayed-out. When the active data object is polygonal, 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 Array Calculator filter can be used to create new arrays by using 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, 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, 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 scalars 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 or an arrow. The glyph can be oriented according to a vector in the input data object, and scaled according to a scalar, 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 plotted.


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 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 less than one will super sample the original data, while sampling rates greater than one will sub sample the data.

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 the center of rotation, reset the center of rotation, show or hide the marker, and edit the center of rotation coordinate value 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.

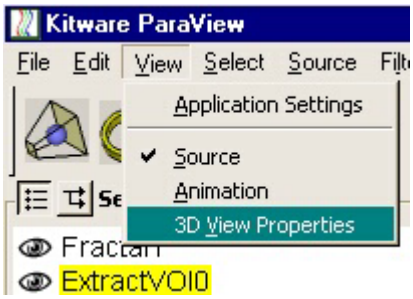

This button can be used to reset the center of rotation to be the center of the currently active data object. This is a useful operation to perform when you want to rotate around a particular data object to view it from all directions.


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 lead to two things. First, the center of rotation marker will become invisible. Second, this button will change appearance to the one shown in the Toolbar at the top of this page (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.

3D View Properties

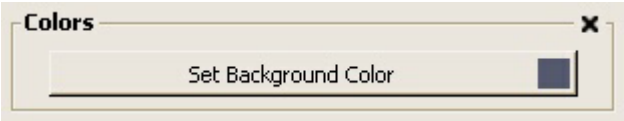
Overview

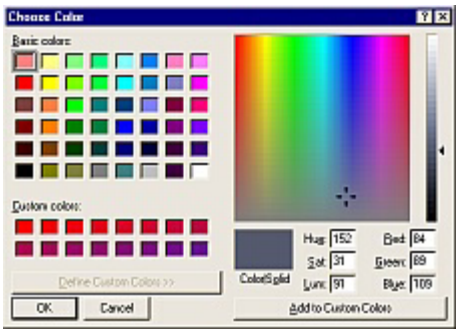


The 3D View Properties option under the View menu on the main Menu Bar can be used to access the property sheet that contains parameters for controlling the camera, setting the background color, selecting advanced rendering parameters and level-of-detail properties, and adding text annotation to the Display Area. This property sheet contains three tabs for General, Annotate, and Camera. The next four sections cover the controls found on the General tab, the section on Annotation covers the Annotate tab, and the final three sections describe the parameters available on the Camera tab.

Background Color

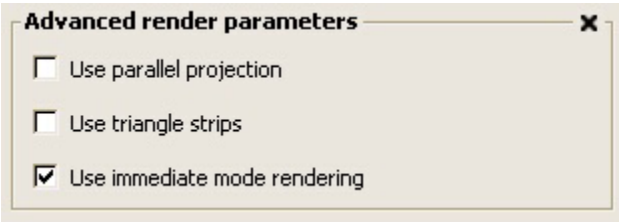
A button for selecting the background color of the Display Area is provided on the General Tab of the 3D View Properties property sheet. This area of the interface will appear as shown below.





Press the button to pop up a dialog from which you can specify a new background color. This dialog is platform specific, and may appear similar to the one shown on the right 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 even if you close ParaView and start again. Changing the background to white is often useful before printing an image.

Advanced Render Parameters



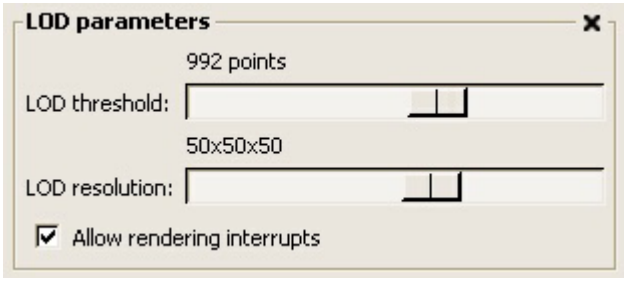
A set of advanced rendering options are provided on the General tab of the 3D View Properties property sheet. This area of the interface will appear as shown on the right. The three check boxes can be used to control global properties of rendering such as parallel versus perspective projection, the use of triangle strips, and immediate mode versus display lists. These options are described below.

Use parallel projection: When checked a parallel camera projection matrix is used for rendering. With a parallel (also referred to as orthographic) projection is used, objects do not get smaller as they get farther away. When this option is not checked, a perspective camera projection is used which is more typical of how we see the world through our eyes.

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 will not necessarily be the same as the one rendered with triangles if this box is unchecked, especially when flat shading is employed. Note that this option will control how data for rendering will be generated in the future, but it will not cause existing pipeline elements to execute again to either produce or eliminate triangle strips.

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. The initial creation of the display list can be slow on some platforms.

LOD Parameters



The controls for the levels of detail used to enable interactive rendering of large data can be accessed in the LOD Parameters area available on 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, with the full resolution data used for generate an image once interaction is complete. ParaView will automatically create the lower resolution representation of your data objects for you to ensure interactivity. The LOD Parameters area will appear similar to the example shown on the right. 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.

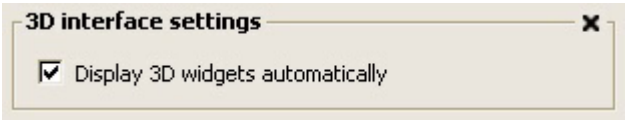
LOD threshold: This slider can be used to control the threshold at which a lower resolution representation of your data is created. For data sets that contain less than the specified number of points, the original data will always be used for rendering. Moving the slider to the left will increase the threshold and will tend to render every data object at full resolution. Moving the slider to the right 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.

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, you can click in the Display Area to begin a new interaction, or click on the interface to change the state of a widget, and the current render will be aborted to respond. When MPI is in use, ParaView uses asynchronous messaging to provide this functionality.

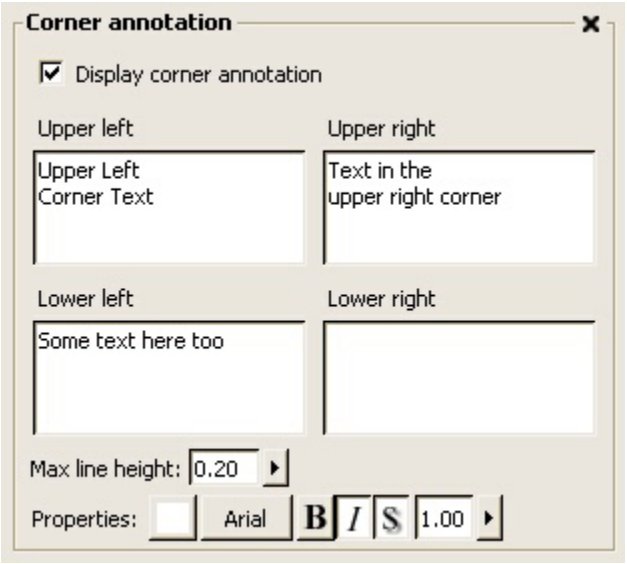
3D Interface Settings



The 3D Interface Settings control area can be 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 will appear 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 as widget for parameter specification. This option can be unchecked if you are working with very large data and plan to specify all parameters manually. This way you can avoid the extra render that occurs to turn on the 3D widget.

Annotation



Text annotation can be added to the four corners of the Display Area using the Corner Annotation area that can be found 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 will appear 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.

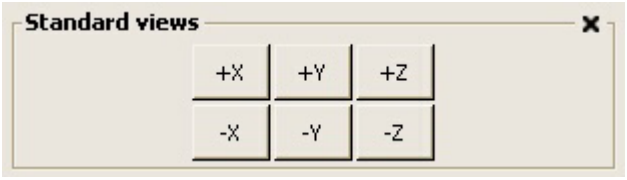
Display corner annotation: When checked the corner annotation is visible.

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.

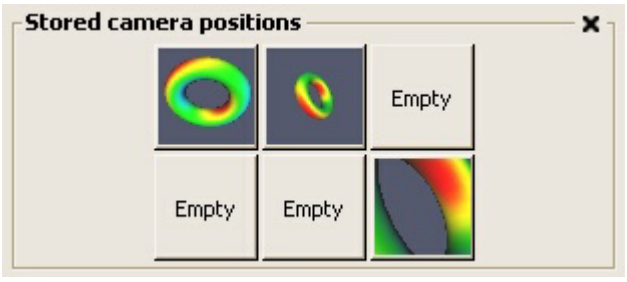
Properties:The first button can be used to specify the color of the text. The second button is used to determine the font family. The next three control whether or not the text is bold, italics, or has a shadow. When the button appears depressed the option is enabled, when the button is raised the option is disabled. In the above example image, the bold option is disabled and the italics and shadow options are enabled. The final control is for the opacity of the text and can range from 0.0 to 1.0.

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 size 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.

Stored Camera Positions



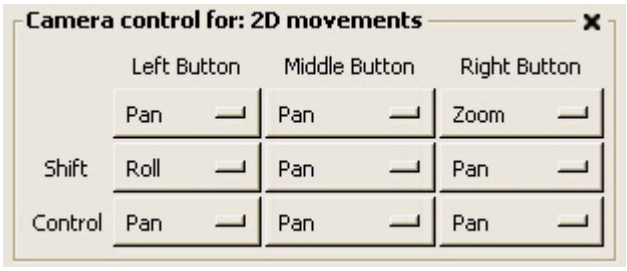
The Stored Camera Positions area can be used to store and retrieve up to six camera viewing parameters, and is located on the Camera tab of the 3D View Properties property sheet. This

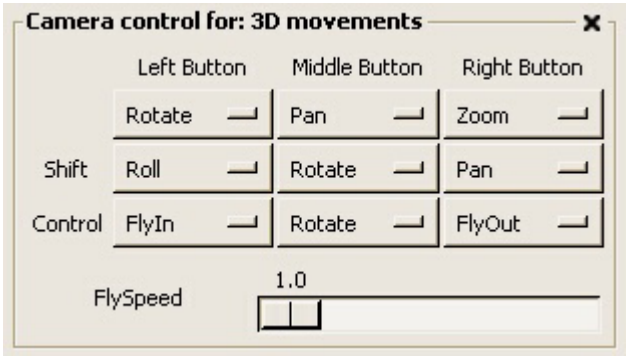
property sheet can be accessed through the View menu on the main Menu Bar. In the example shown on the right you can see that three camera positions have been saved, and three buttons are still empty. To store the current camera parameters in one of the six locations, right click on that button. To retrieve the saved position, left click on the button.

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 will 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 these options 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 on 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. The speed at which the camera moves is control by the Fly Speed scale. 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 indicate 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.

Display Properties

Overview

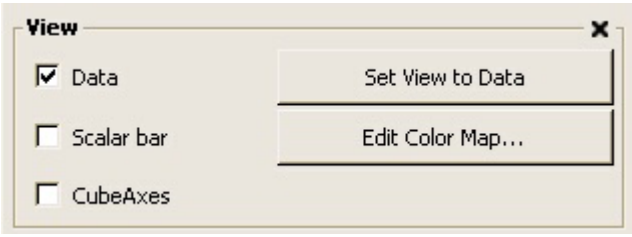
When a data object is created by loading data, creating a source, or applying a filter, a property sheet is associated with that data object. To view this property sheet, select the data object from the list of objects in the Select menu on the main Menu Bar, or select the item from the Selection / Navigation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

View Settings

The top region on the Display tab of the property sheet associated with an item in ParaView contains the View settings. The appearance of this regions depends on the setting of the Color By option in the Color area. Two possible configurations are given below. The top one is the state of this area when the data is color by property, and the bottom shows the configuration when the data is colored by some scalar or vector component of the data.



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

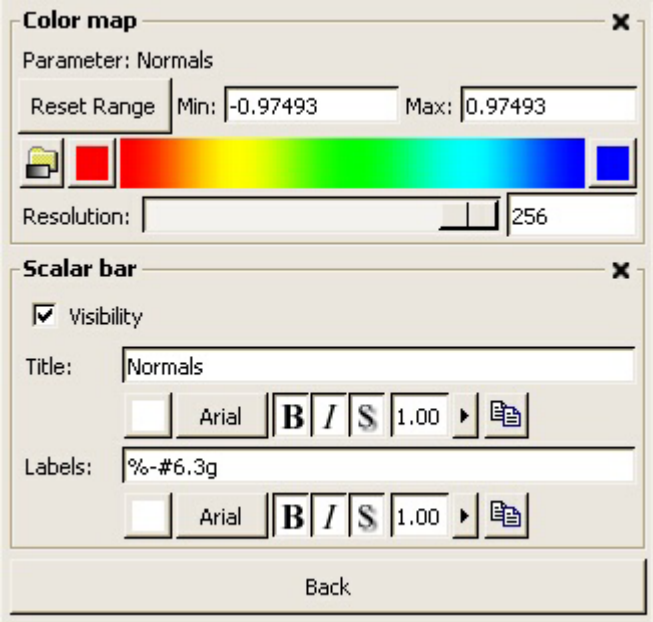
Data: This check box can be used to toggle the visibility of the data. This visibility can also be controlled using the eye icon in the Selection Window.

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 displayed 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 tic 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. Additional controls for the title and labels of the scalar bar are provided on the property sheet displayed when Edit Color Map is selected, and these are described below.

CubeAxes: Toggle the visibility of the labeled axis displayed along three outer edges of the bounding box of the data object.

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.

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



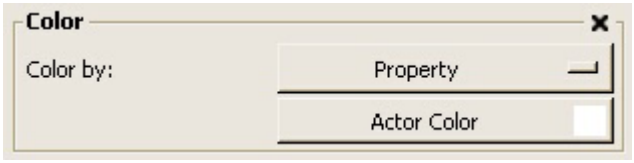
The top region of this property sheet allows you to adjust the color map used to map scalar values to color. The indicated range is the initial range computed for this data object. If you have adjusted parameters that have changed the scalar range you will either need to press Reset Range to reset to the full scalar range, or to 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.

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 tic labels. In addition, you can set the color, font family, attributes such as 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, you will press the  button at the end of the text customization buttons in the Labels area.

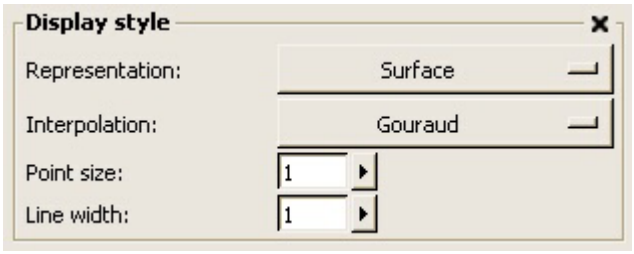
Pressing the Back button will return you to the Display tab of the data object's property sheet.

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 on the right. In the example shown the Color by selection has been set to Property, and an Actor Color button is displayed allowing you to choose a new solid color for this actor. 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 disappear, and the Edit Color Map option will appear in the View region of this property sheet.

Display Style



On the Display tab of the property sheet that is 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.

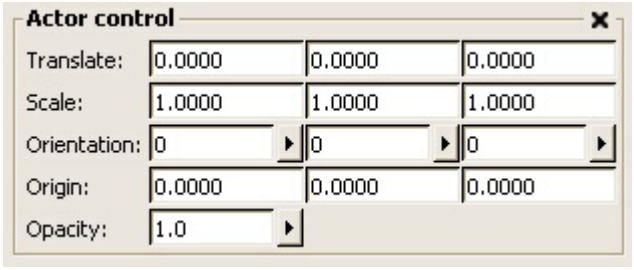
Representation: You can choose between a surface, a wireframe or a point representation of the data. When a surface representation is selected, primitives such as polygons and triangles are drawn as filled geometric primitives. When wireframe mode is selected, only the edges are drawn. In point mode, only the points defining the geometry are displayed.

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) then this slider can be used to control the thickness of the lines.

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 looses focus (for example when you click in another text entry area).

Translate: These three values represent a translation in X, Y, and Z.

Scale: Enter scale values along the X, Y, and Z axes.

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.

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.

Data Information

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.

Statistics

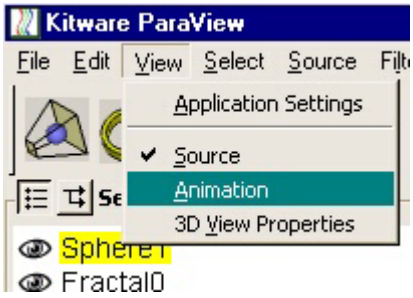
The top region on the Information tab of the property sheet associated with a data object will provide basic statistics about that data object including the type of data, the number of cells, and the number of points. An example is shown on the right for polygons (vtkPolyData) data.

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.

Animation

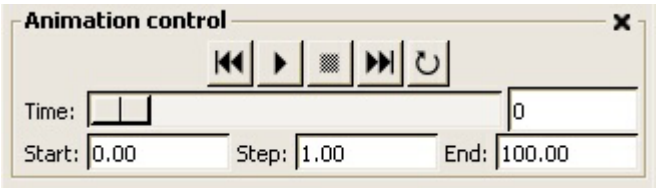
Overview



ParaView provides some basic animation controls that can be accessed via the Animation property sheet. To bring up this property sheet, select the Animation option from the View menu on the main Menu Bar as shown in the image on the right.

Animation is an advanced feature of ParaView which typically requires scripting. Some basic details are provided in this chapter. A future edition of the guide will provide a more detailed description of the animation facilities of ParaView.

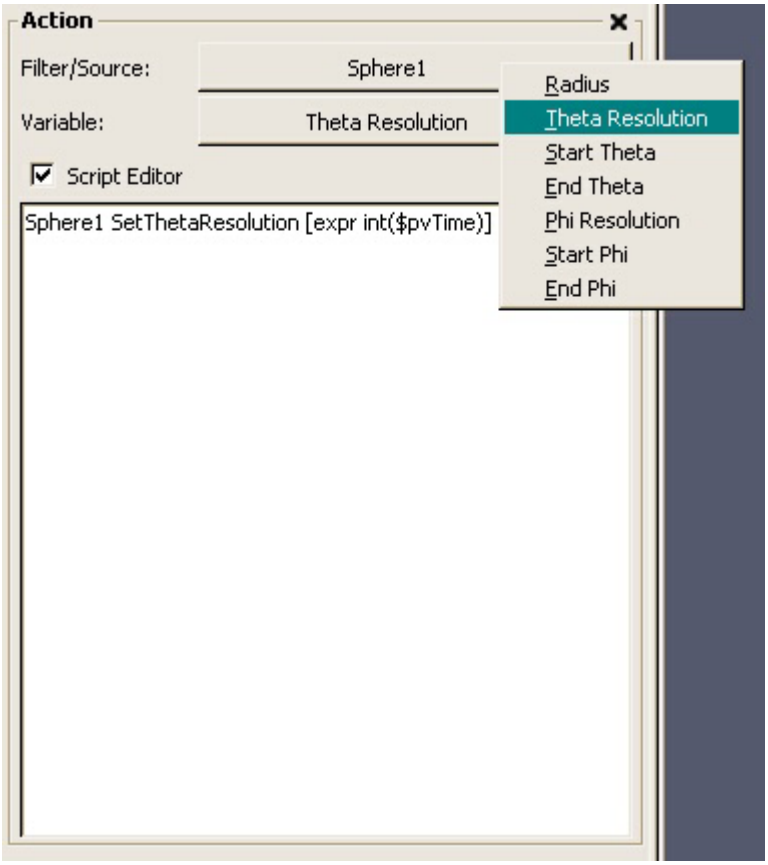
Animation Control



Once an animation has been defined in ParaView, the Animation control area can be used to play through the frames of the animation. The top set of buttons work similar 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 enable a looping mode. The Time slider can be used to jump to a particular frame. The Start, Step, and End values define the length and resolution of the animation.

Action

The Action area on the Animation property sheet is where the animation is defined. By checking the Script Editor option you can see a text entry box with the current script in there. Selecting a Filter or Source from the list will provide you with a list of variables that can be controlled in the animation. In the example shown below we are going the animate the value of Theta Resolution.



Saving Results

Overview

Several type of data can be saved from ParaView. The currently selected data object can be saved as a VTK file. A session file can be saved capturing the all states of the system since startup. Finally, and image of the Display Area can be saved.

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 VTK file. You can choose between an "old" VTK file (*.vtk extension) or a new format with an extension based on the data type.

ParaView Session Files

Selecting the Save Session option from the File menu will save the state of the system as a history of operations performed since ParaView was started. This is not an efficient means of saving the state of ParaView, but it is the only method available in this release.

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 images (*.jpg), PNG images (*.png), binary PPM images (*.ppm) and tiff images (*.tif).

Copy And Print

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 option or control-v (as supported by the application). This functionality is available only on Windows platforms.

Image Print

The Print option on 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 include 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.

Overview

Depending on which platform you are using, ParaView may supply 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.