Introduction to OpenDX

Mike Bailey

Oregon State University

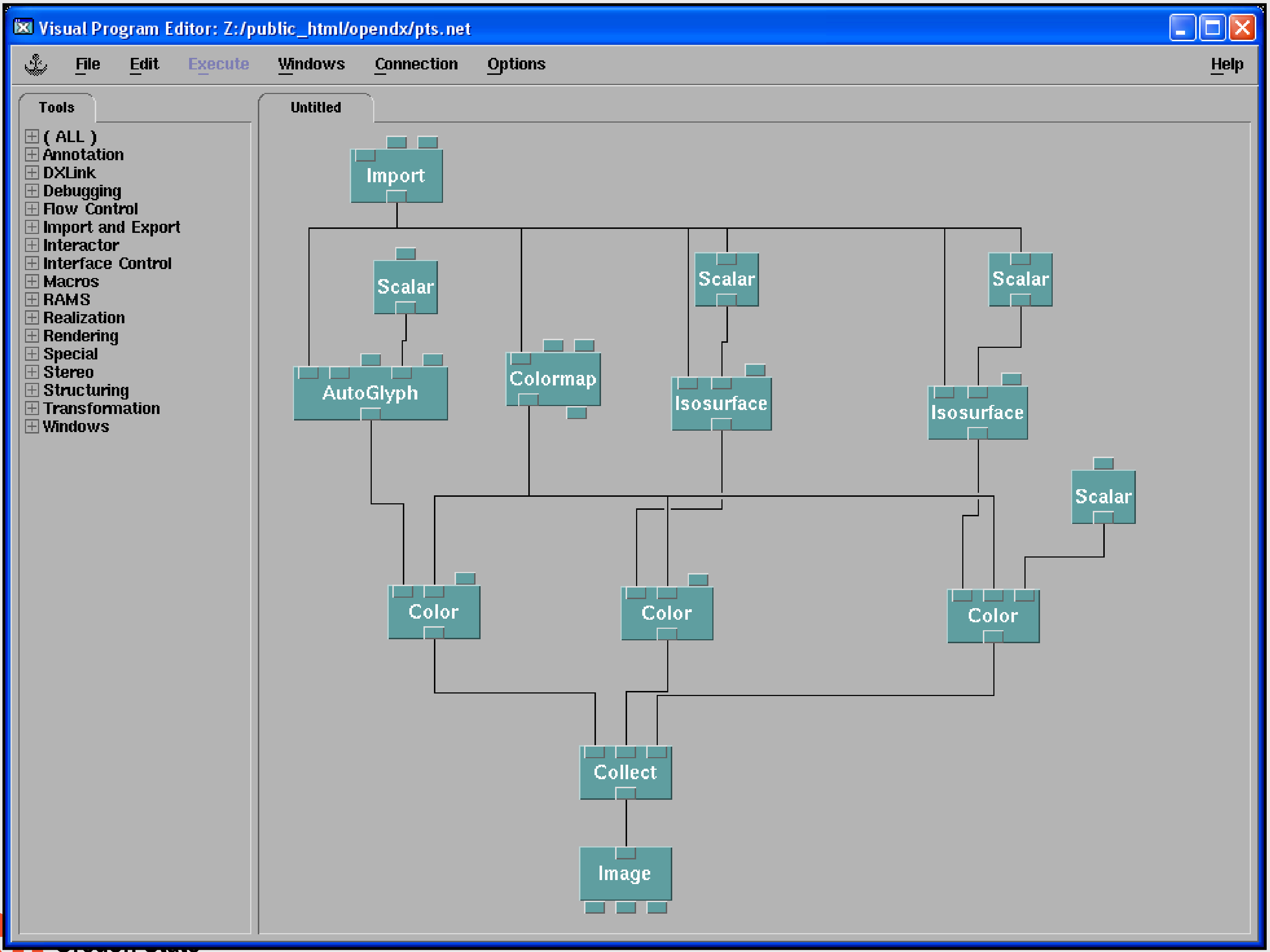
OpenDX

- Started out life as *IBM Visualization Data Explorer*
- When the product was cancelled, IBM put it into Open Source and renamed it *OpenDX*
- Basic premise is a series of interconnected modules, living together in an environment called the Visual Program Editor (VPE)
- There are lots of provided modules
- You can also write your own

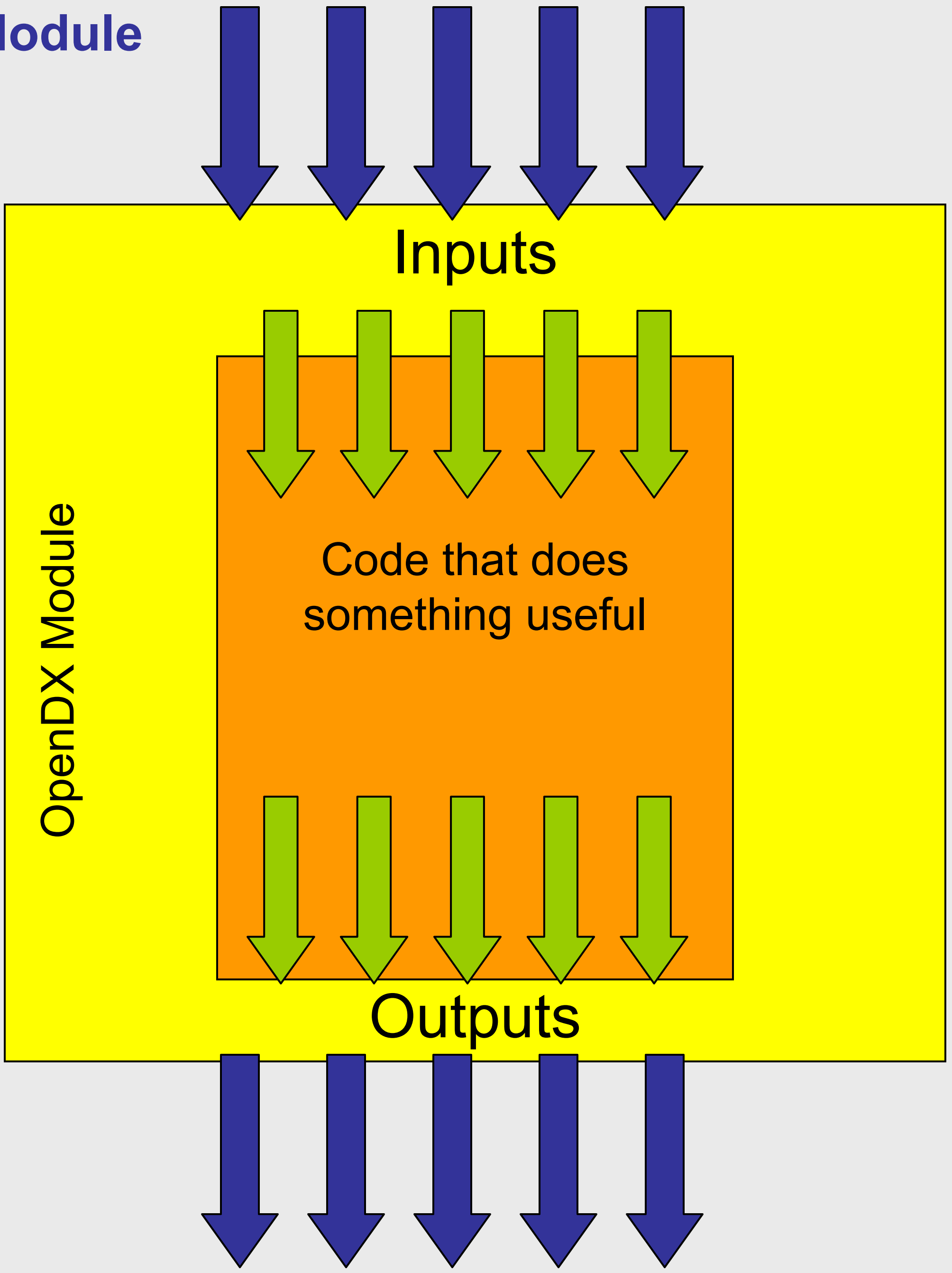
`http://www.opendx.org`

`http://www.vizsolutions.com`

`http://eecs.oregonstate.edu/~mjb/opendx`


Visual Program Editor: Z:/public_html/opendx/pts.net
File
Edit
Execute
Windows
Connection
Options
Help
Tools
(ALL)
Annotation
DXLink
Debugging
Flow Control
Import and Export
Interactor
Interface Control
Macros
RAMS
Realization
Rendering
Special
Stereo
Structuring
Transformation
Windows
Untitled
Import
Scalar
Scalar
Scalar
AutoGlyph
Colormap
Isosurface
Isosurface
Scalar
Color
Color
Color
Collect
Image

The Structure of an OpenDX Module



Oregon State University

Steps in Creating a Visualization

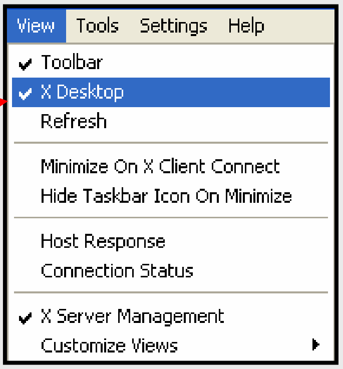
Seven Steps to Creating a Visualization

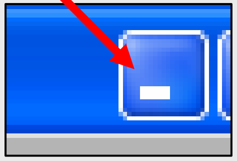
1. **Get the data**
2. **Formulate a scientific strategy. What do you want to show? How do you want to show it?**
3. **Import the data**
4. **Create a *simple* OpenDX network**
5. **Incrementally embellish the network. Save it often!**
6. **Choose what quantities you want to interact with. Change the Interactor styles to match the quantities being modified.**
7. **Create the output.**

Starting OpenDX in OSU's Computer Graphics Education Lab

Starting OpenDX on the OSU CGEL Systems

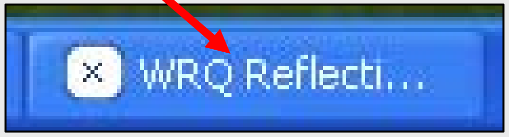
1. Start → All Programs → WRQ Reflection → Reflection X
2. In the *View* menu, click off *X Desktop*
3. Minimize the Reflection X window (the _ in the upper right corner)
4. Start → All Programs → OpenDX → DX





Quitting OpenDX on the OSU CGEL Systems

1. Select *Quit* from the OpenDX Main Menu
2. Maximize the Reflection X window by clicking here in the Task Bar
3. In the *File* menu, select *Exit*

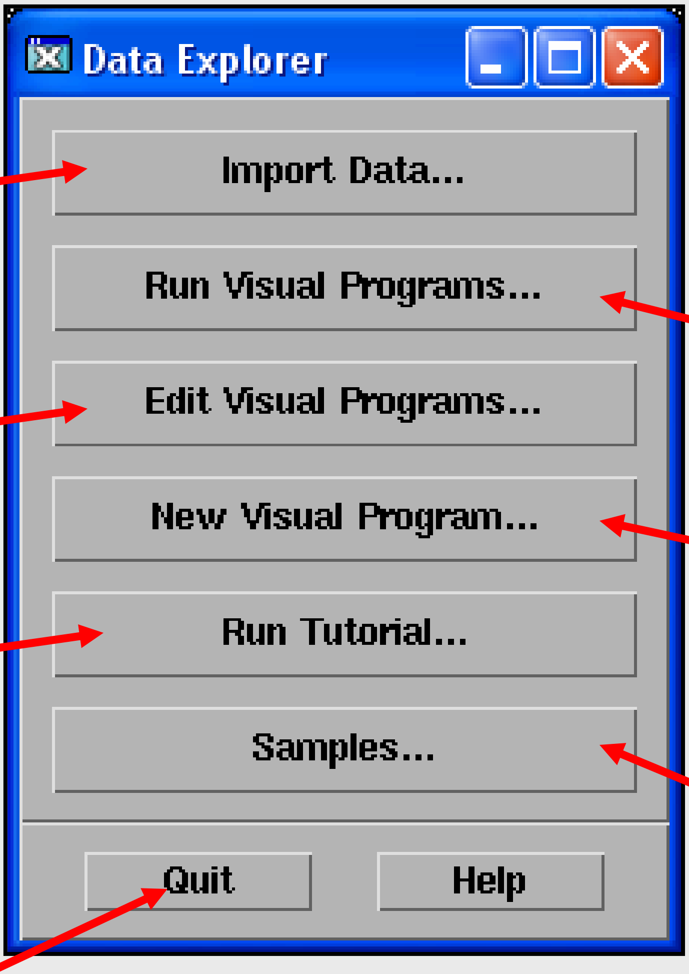


Oregon State University

mjb – November 27, 2006

The OpenDX Main Menu and Categories of Modules

The OpenDX Main Menu



Get into the Data Prompter program

Run an OpenDX network without seeing the network

Run an OpenDX network and be able to edit the network

Create a new OpenDX network

Run the internal OpenDX tutorial

Load, and be able to edit, one of the OpenDX sample networks

Exit OpenDX

Nine Categories of OpenDX Modules

Annotation	Interactor	Special
Debugging	Realization	Structuring
Import & Export	Rendering	Transformation

Annotation OpenDX Modules

- AutoAxes – creates an axis box for whatever data you are plotting
- AutoGlyph – designs and produces glyphs for the data based on the data values
- Caption – creates caption text for an image
- ColorBar -- creates a colorbar to be displayed
- Format – creates a string from a number (used to create file names)
- Glyph – produces an identical glyph for every point in the data
- Legend – produces a legend to be displayed
- Plot – creates a 2D plot
- Ribbon – creates a flow field ribbon
- Text – displays text in 3D space
- Tube – creates a flowfield tube

Debugging OpenDX Modules

- Describe – describes an object
- Print – prints information about a field to the Message Window

Import & Export OpenDX Modules

- Export – writes data from OpenDX into a file
- Import – reads data into OpenDX from a file
- ImportSpreadsheet – reads data into OpenDX from a tabular file
- Include – includes or excludes points in a field based on their data values
- ReadImage – reads an image into OpenDX from a file
- Reduce – filters and resamples a field into a lower resolution
- Refine – interpolates a field into a higher resolution
- Slab – takes a positional subset of the data
- Slice – takes a positional slice through the data
- WriteImage – writes an image from OpenDX into a file

Interactor OpenDX Modules

- FileSelector – presents a dialog box to let you select a file
- Integer – allows the user to input an integer number
- Scalar – allows the user to input a floating point number
- Selector – allows the user to select one of a number of options
- String – allows the user to input a string
- Toggle – allows the user to select one of two options
- Vector – allows the user to input a vector

Realization OpenDX Modules

- AutoGrid – maps a set of scattered points onto a grid
- Band – divides a field into bands
- Connect – creates triangle connections for scattered data points in a field
- IsoSurface – creates surfaces or lines of constant data value
- MapToPlane – projects a data field onto an arbitrary plane
- RubberSheet – deforms a surface field by the amount of the data value at each point
- ShowBox – creates a bounding box for display
- ShowConnections – displays the outline of connectivity elements in a field
- ShowPositions – displays the positions in a field
- Streakline – computes an advection path through a changing flow field
- Streamline – computes a path through a non-changing flow field

Rendering OpenDX Modules

- AmbientLight – specifies the ambient light
- Arrange – creates a single side-by-side image from a collection of images
- AutoCamera – selects a good camera view of the data
- Camera – specifies a camera view
- Display – a more elaborate image-rendering system than Image
- Image – renders and displays field data
- Light – specifies a distant (parallel) light source
- Normals – compute point or face normals for shading a surface
- Render – renders a field and creates an image
- Rotate – rotates field data
- Scale – scales field data
- Shade – specifies object-shading parameters
- Transform – performs a general matrix transform of an object
- Translate – translates field data

Special OpenDX Modules

Colormap – presents an interactive tool for specifiying color vs. data value

Receiver – receives the output of a Transmitter

Sequencer – creates an animation “VCR” display

Transmitter – “wirelessly” connects a network to a receiver

Structuring OpenDX Modules

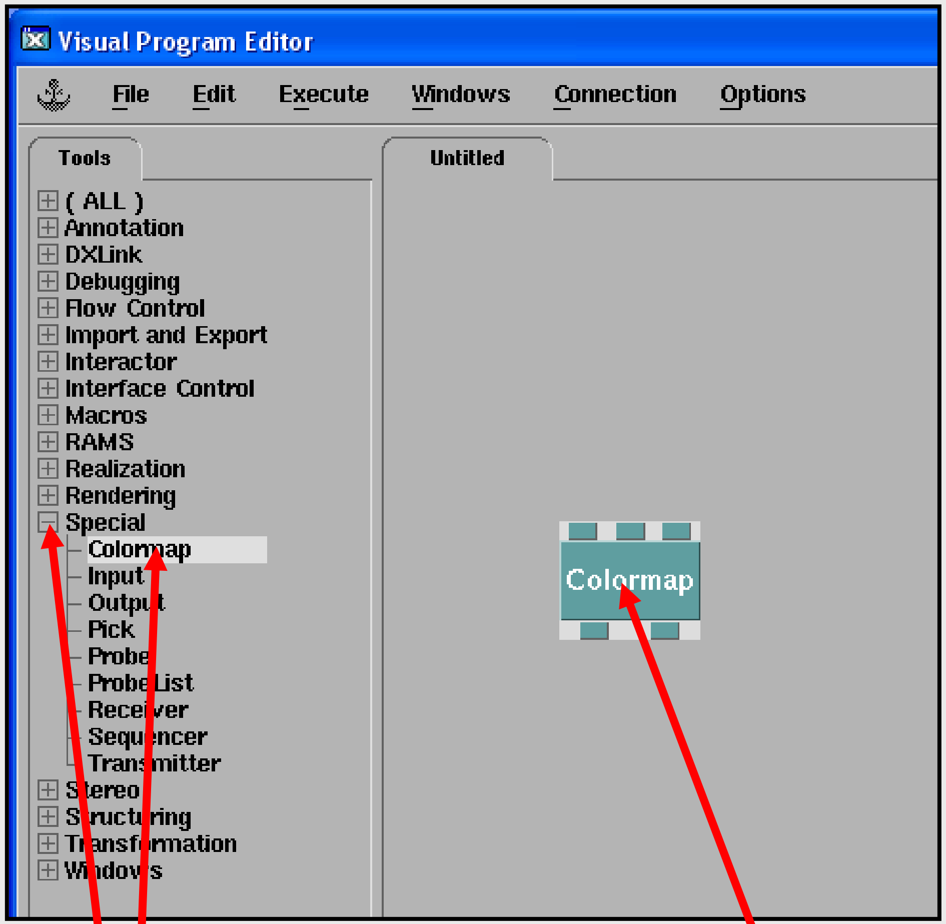
- Collect – collects objects into a group
- Inquire – returns information about a field
- Mark – marks a new field component as “data” (e.g., for Compute)
- Remove – removes a specified component from a field
- Rename – renames a specified component in a field
- Unmark – undoes the effects of Mark

Transformation OpenDX Modules

- AutoColor – automatically color a data field (blue→green →red)
- Color – assign a color by name of by RGB values
- Compute – perform point-by-point arithmetic on a field’s “data” component
- DivCurl – computes the divergence and curl of a flow field
- Equalize – apply histogram equalization to a field
- Gradient – computes the gradient of a scalar field
- Histogram – creates a histogram that can be rendered with Plot
- Map – projects one field’s data onto another field’s geometry
- Measure – calculates surface area and volume of a geometry (e.g., isosurface)
- SimplifySurface – reduces the size of the triangular mesh
- Statistics – computes the mean, standard deviation, variance, minimum, and maximum of a field’s data

Adding and Connecting Modules

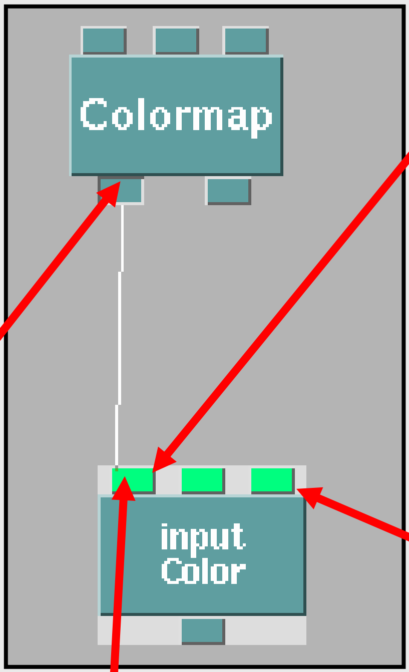
Adding a Module into the Visual Editing Area



It's not drag-and-drop, it's click-and-click

1. Left-click on the module category to list its modules.
2. Left-click on the module you want to add
3. Move the cursor into the Editing Area and left-click

Connecting Modules in the Visual Editing Area



If an input tab is in the "up" position, you are allowed to try to connect to it.

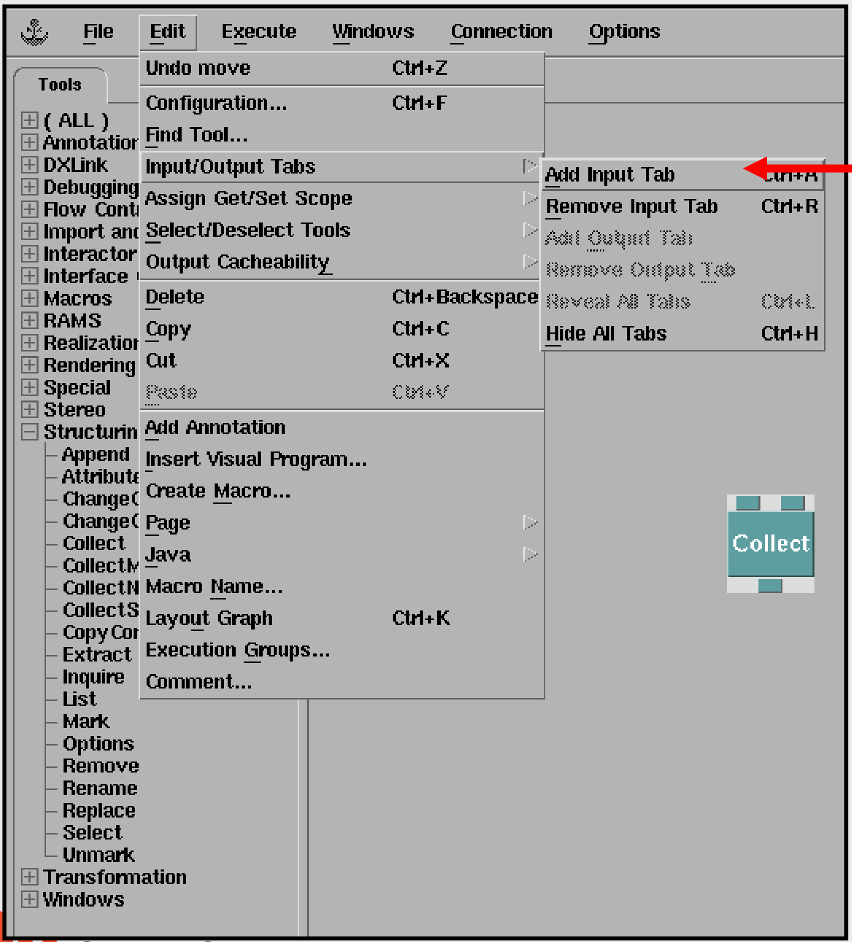
If an input tab is "down", then it has already been set to a constant within the module itself, and cannot take an external connection until that constant has been un-set.

Just because an input tab is up, however, doesn't mean that this input is data-compatible with the output you are trying to connect to it. Data-compatability is indicated by the input tab(s) turning bright green.

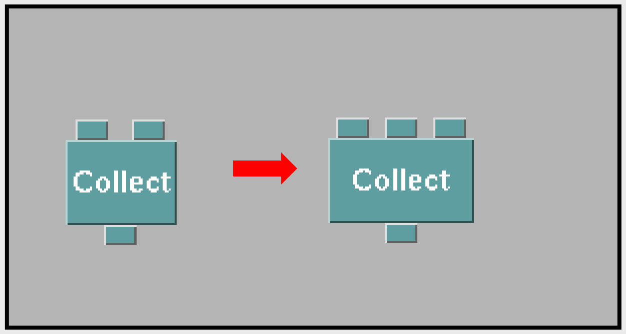
This, however, still doesn't imply that the connection makes logical sense. ☺

1. Left-click on the output tab of the module you are connecting from
2. Keeping the left button down, drag to the input tab of the module you are connecting to
3. When you get close, the tabs to which a connection make sense will highlight in green
4. Move the cursor on top of the tab you want to connect to, and release the left mouse button
5. To disconnect, reverse the process. Click on the input tab and drag back to the output tab.

Some Modules Can Have Variable Numbers of Tabs



Edit→Input/Output Tabs→Add Input Tab

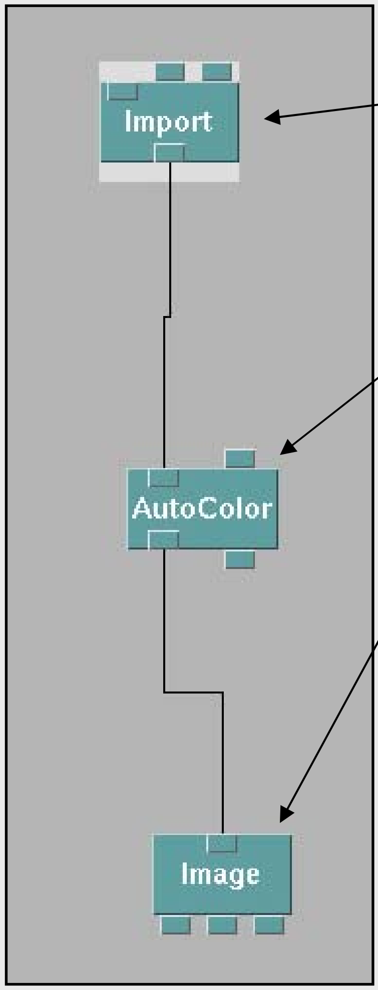


Collect and *Compute* are two common modules that work this way

Oregon State University

Terrain Visualization

Terrain Visualization

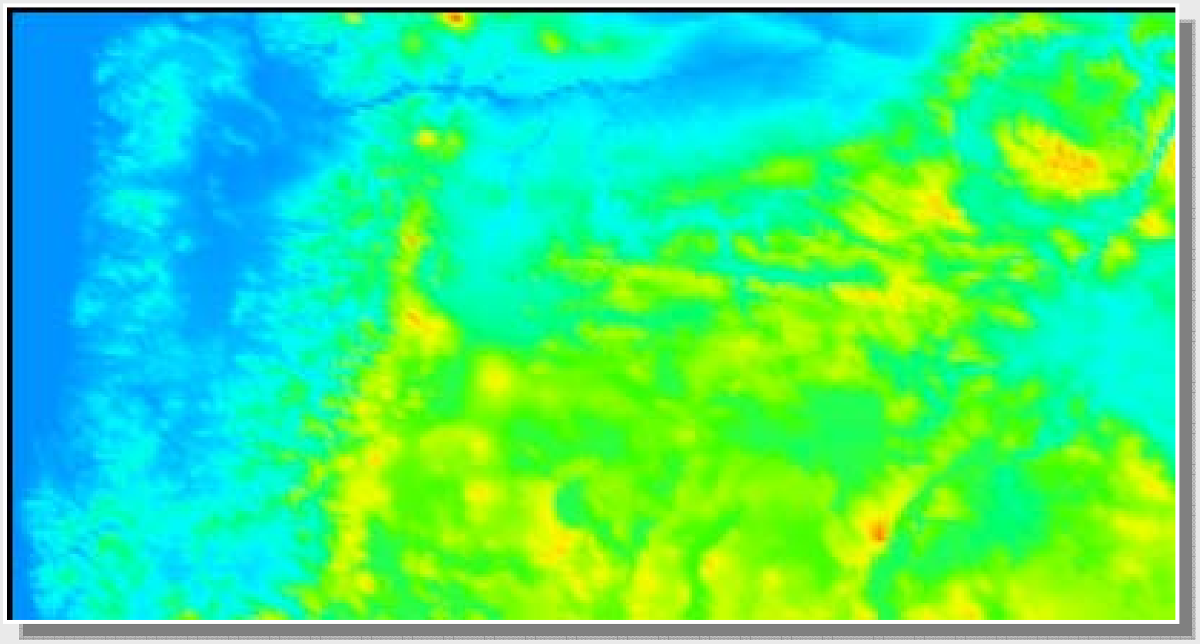


Import/Export

Transformation

Rendering

Start simply, then embellish !

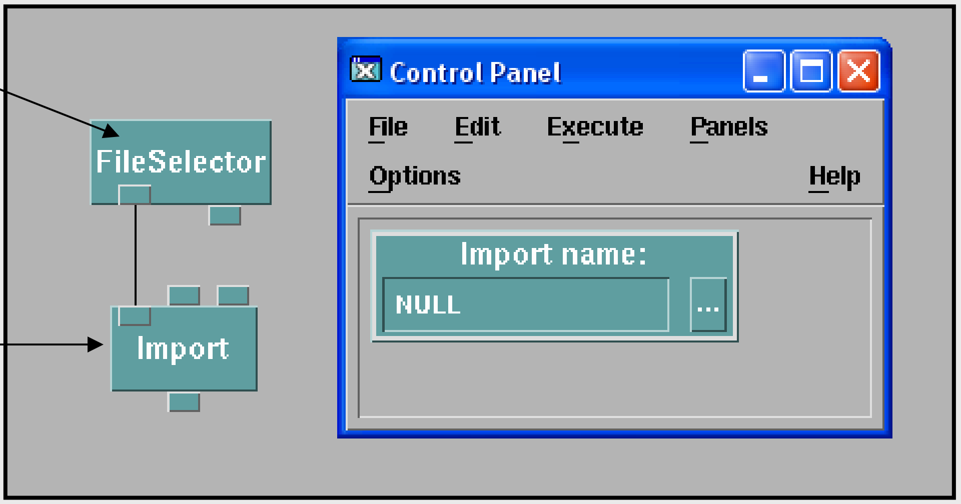


OSU Oregon State University

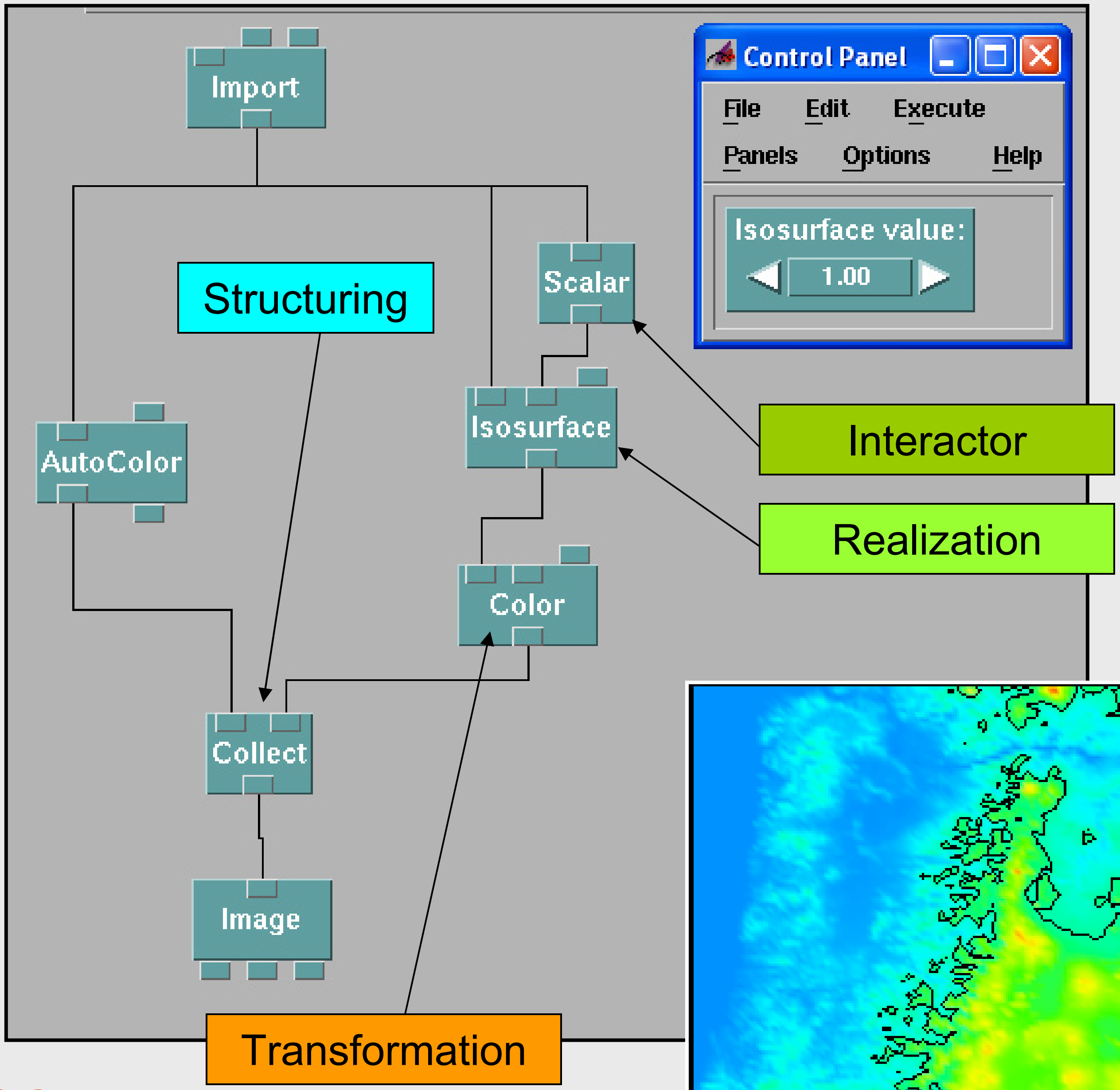
The Import and FileSelector Modules

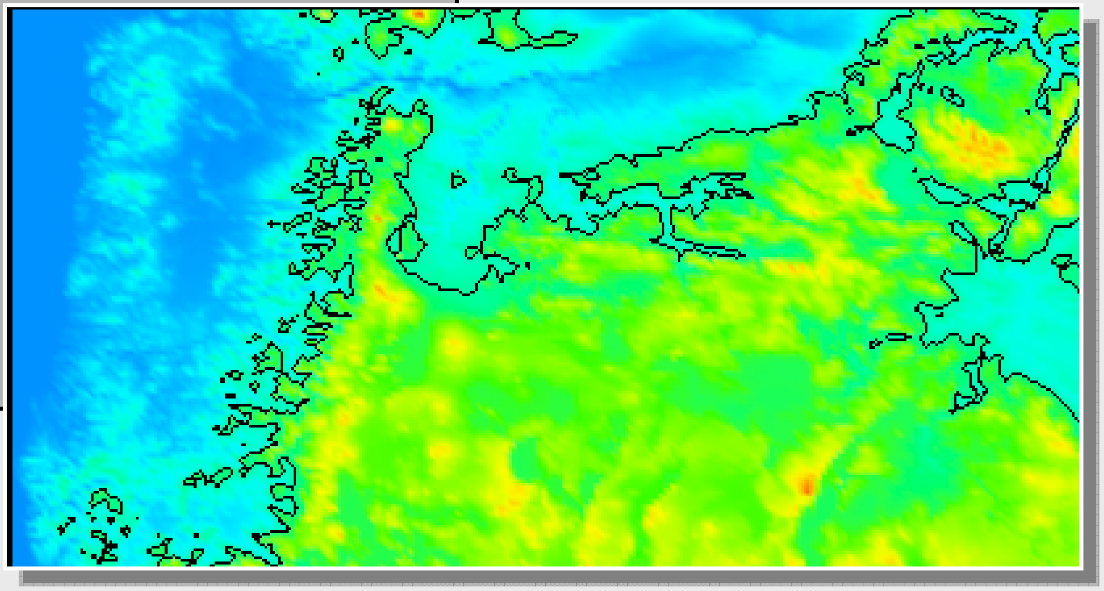
Interactor

Import/Export



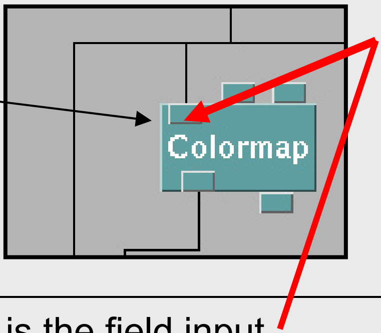
You can type a filename into the Import module, but hooking in a FileSelector module makes it way easier and friendlier

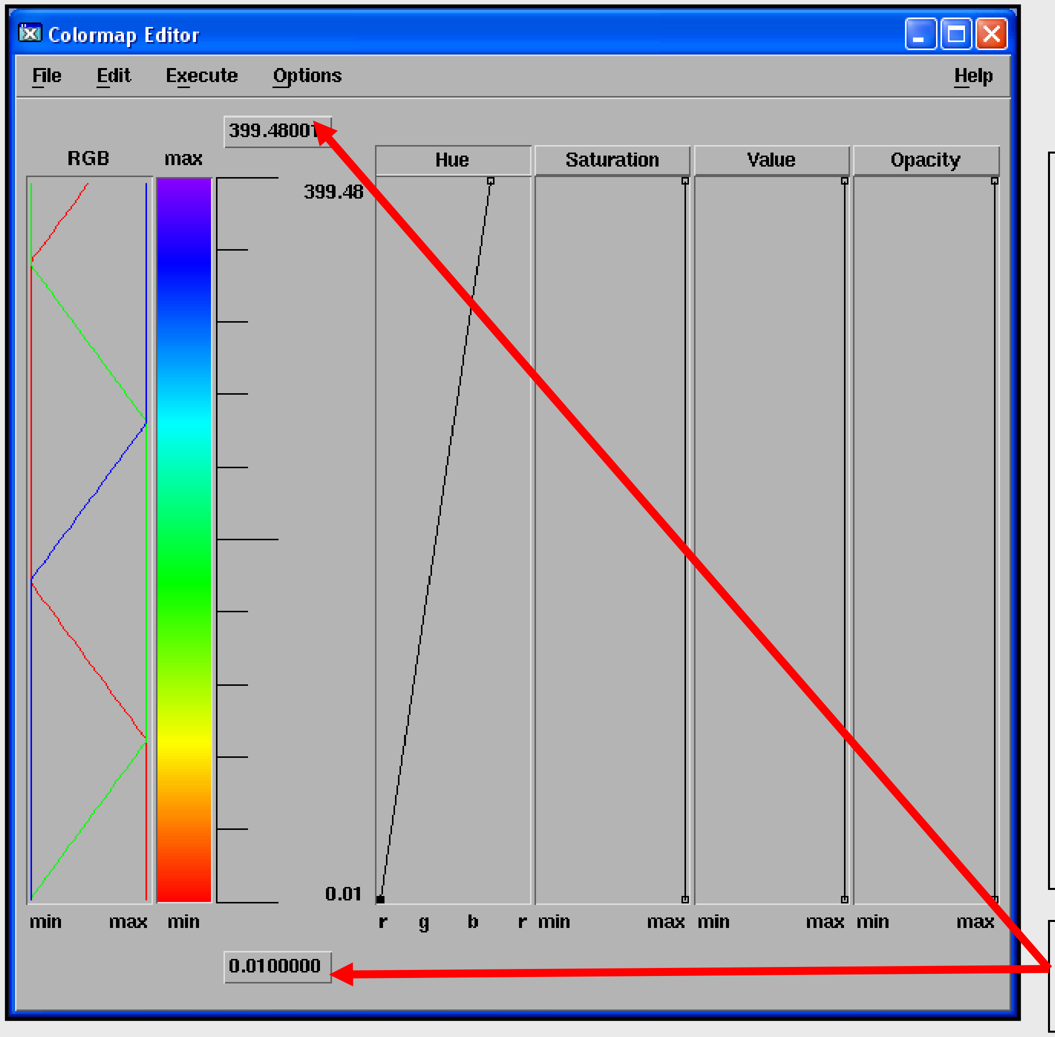

Import
Structuring
Scalar
Control Panel
File
Edit
Execute
Panels
Options
Help
Isosurface value:
1.00
Isosurface
Interactor
AutoColor
Realization
Color
Collect
Image
Transformation



The Colormap Editor Module

Special





• The first input "tab" is the field input.

• Click on the Hue, Saturation, Value, or Opacity labels to edit that curve.

• Double-click on a line to add a control point there.

• Click on a control point to select it.

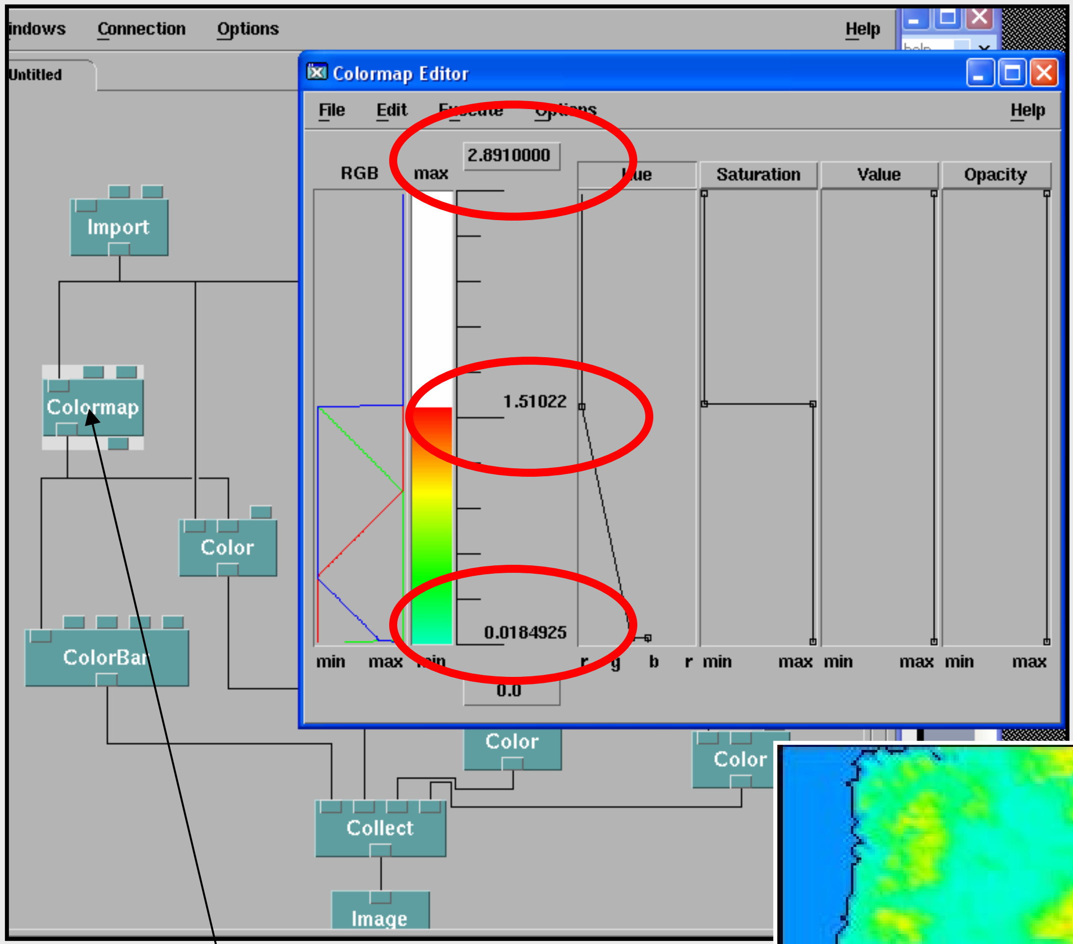
• Sweep a box over several control points to select them all.

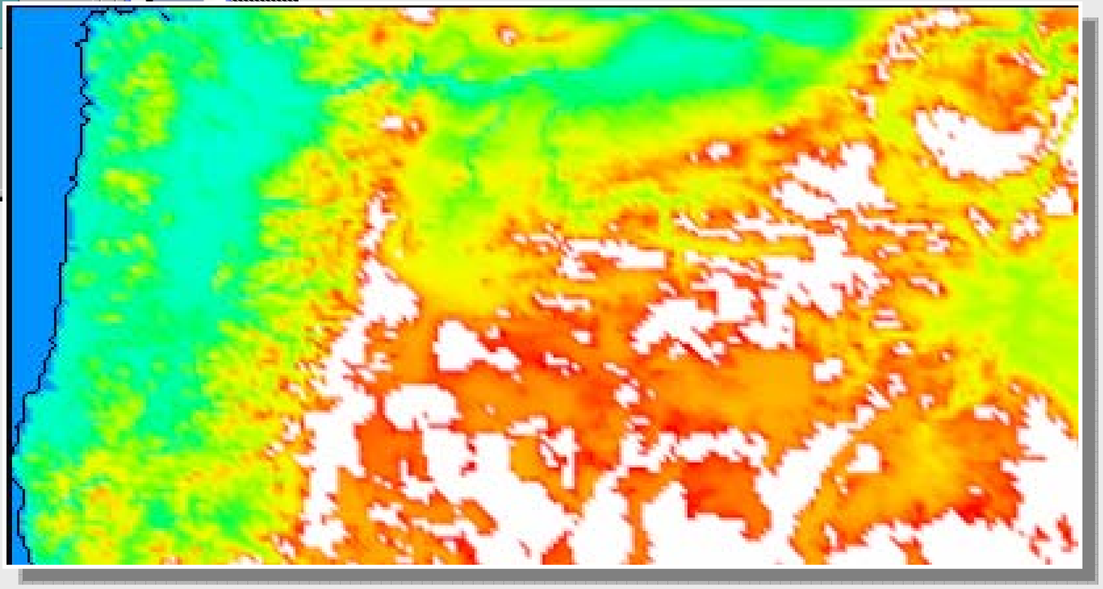
• Hold down the left mouse button on a control point to move it. If several are currently selected, all will move together,

• Edit → Delete to remove selected control point(s).

The data value range over which the colors apply is determined by scanning the data itself.

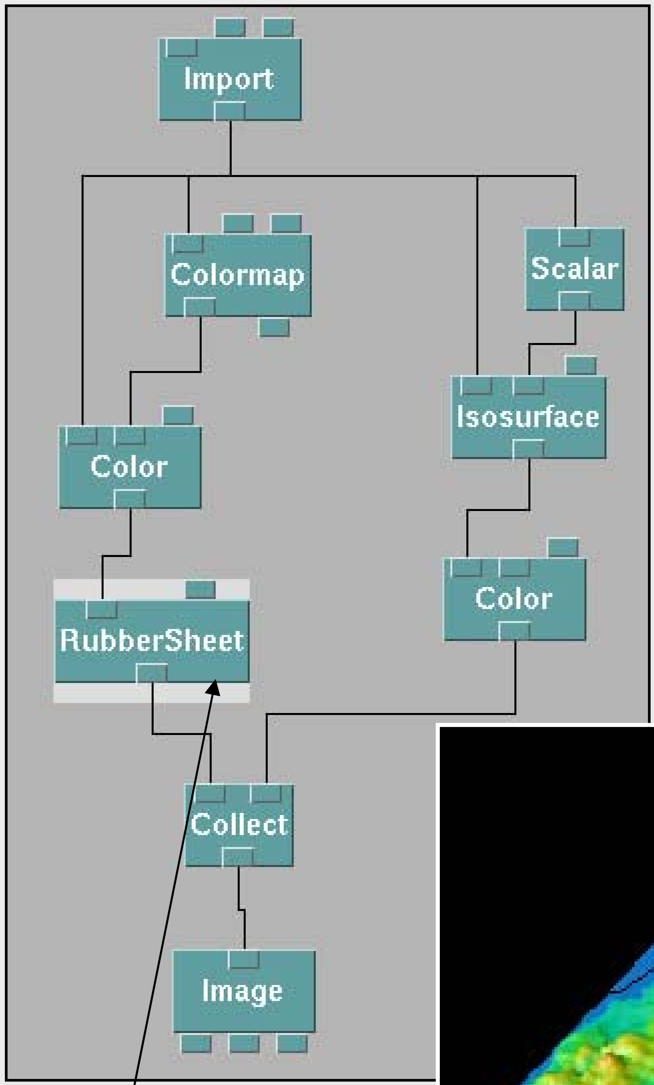
The Colormap Editor in Action

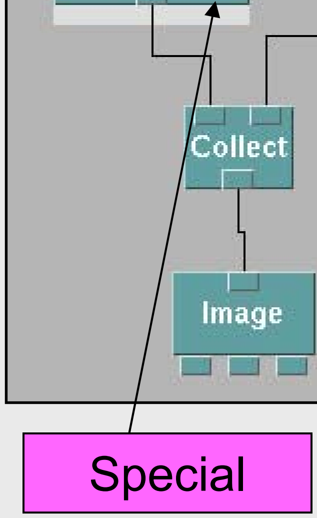


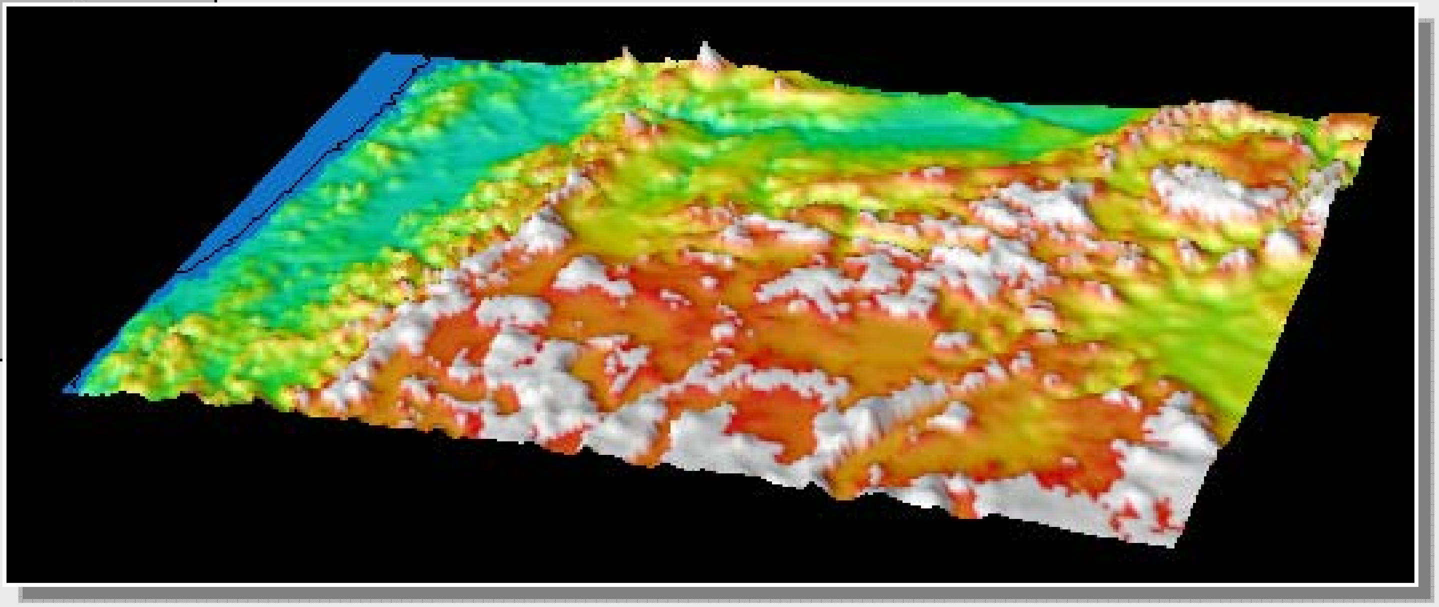


Special

Rubbersheeting the Terrain Surface



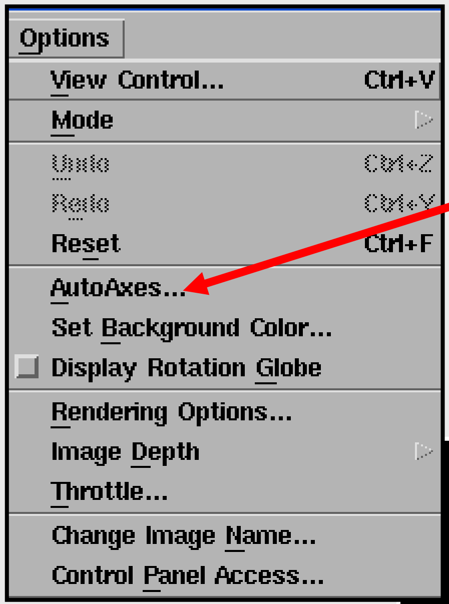




Oregon State University

The Image Window

Image Window Options



The *AutoAxes* option has many ways to embellish the visualization with axes, labels, grids, etc.

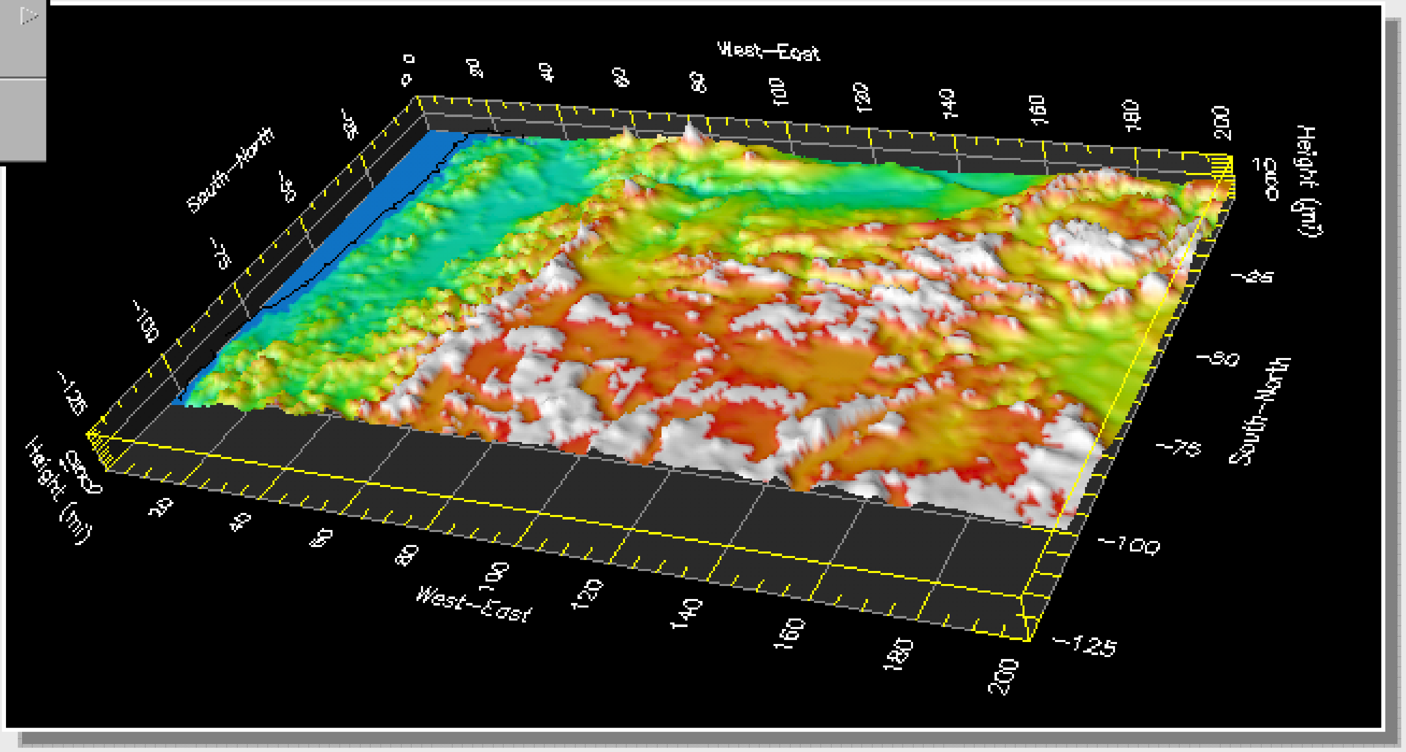


Image Window Options

The *Mode* option lets you set what scene transformation the mouse will perform.

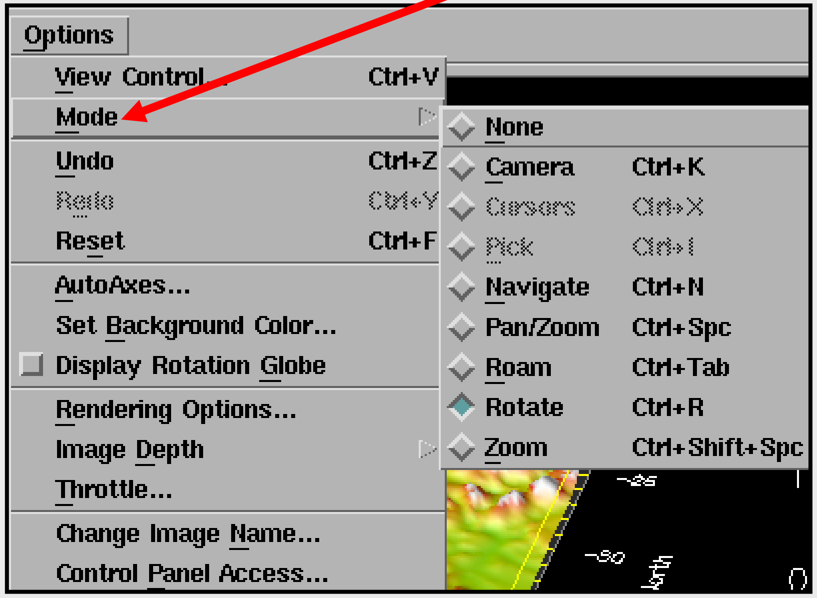
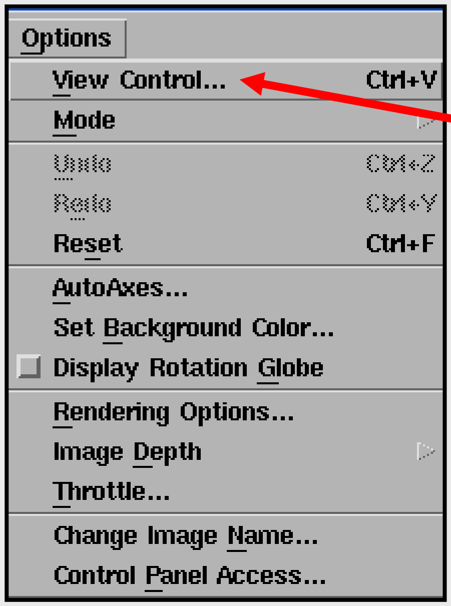
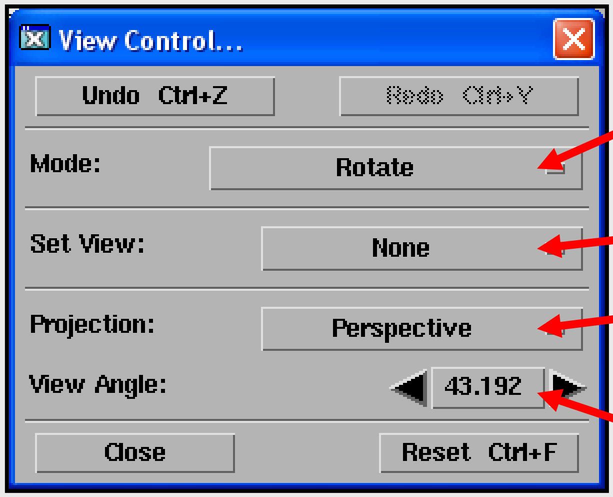


Image Window Options



The *View Control* option lets you set various aspects of how the scene will appear.



Mode: Rotate
Set View: None
Projection: Perspective
View Angle: 43.192

Same as the *Mode* option

Set a pre-defined view

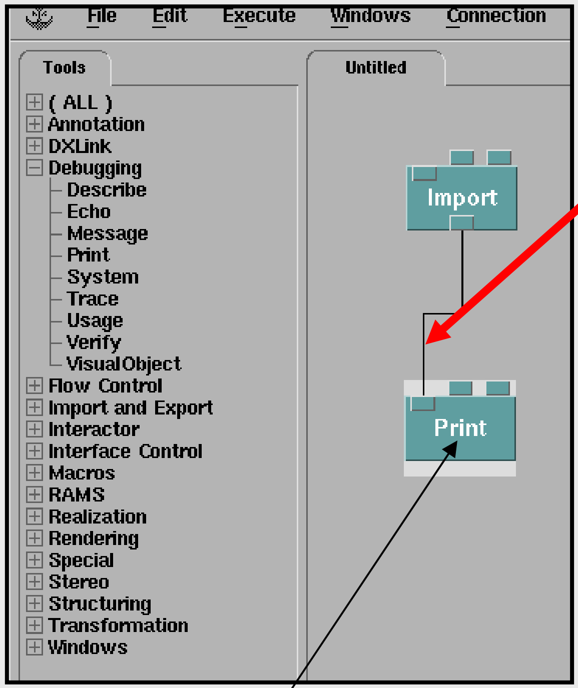
Specify Perspective or Orthographic 3D projection

If using Perspective, this specifies the field-of-view angle. The larger this number, the more severe the perspective will be.

Oregon State University

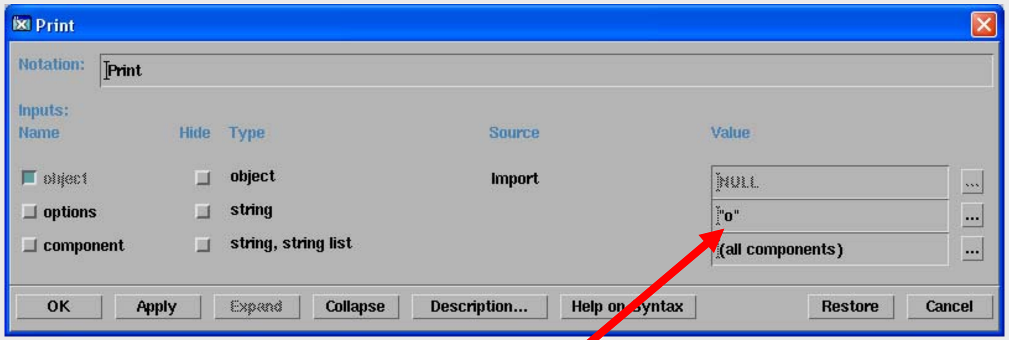
Debugging

The Print Module



Debugging

First argument is the field to print



Notation: Print

Inputs:

Name	Hide	Type	Source	Value
object		object	Import	NULL
options		string		"o"
component		string, string list		(all components)

Second argument is a string with one or more characters:

- **r** recursively traverse the object
- **o** print only the top level of the object
- **d** print first and last 25 items in arrays, as well as headers
- **D** print all the items in arrays as well as headers
- **n** print object to **n** levels.

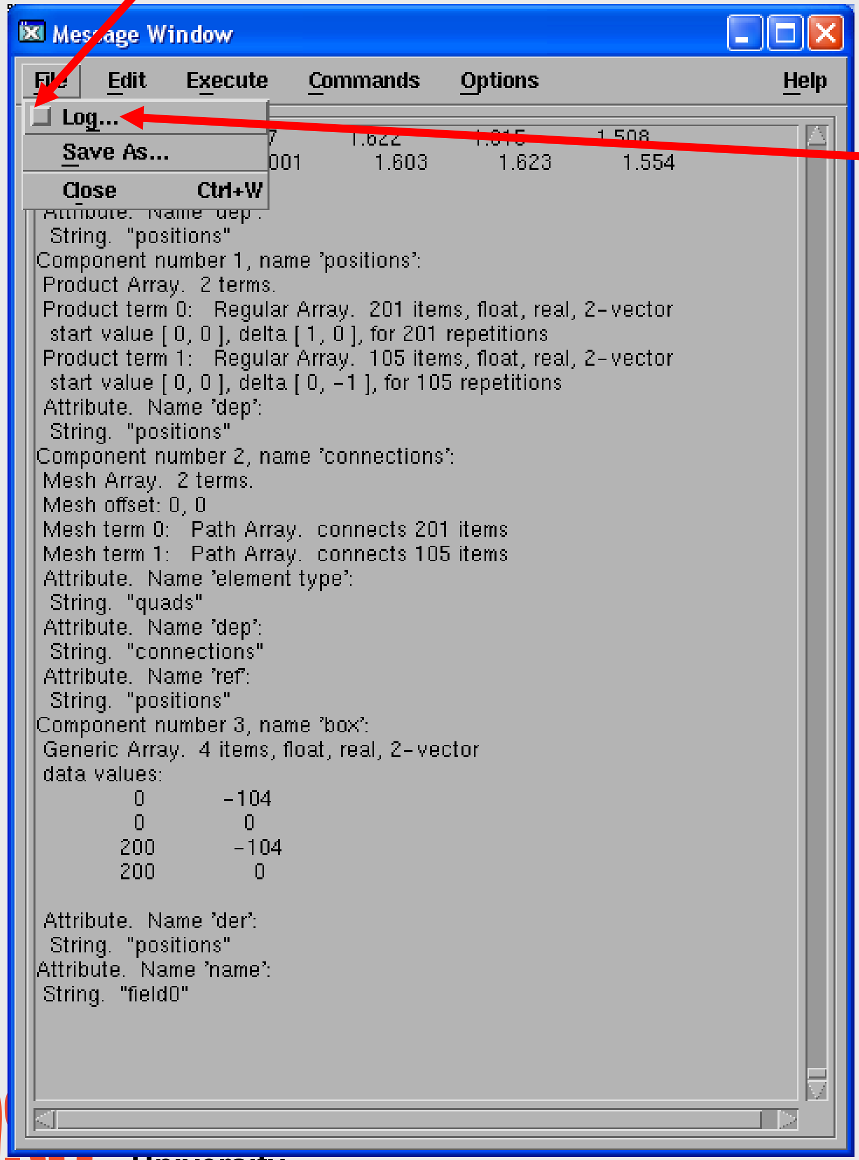
"rD" works well

Oregon State University

Printing to a File ("logging")

Checkbox to turn logging on/off

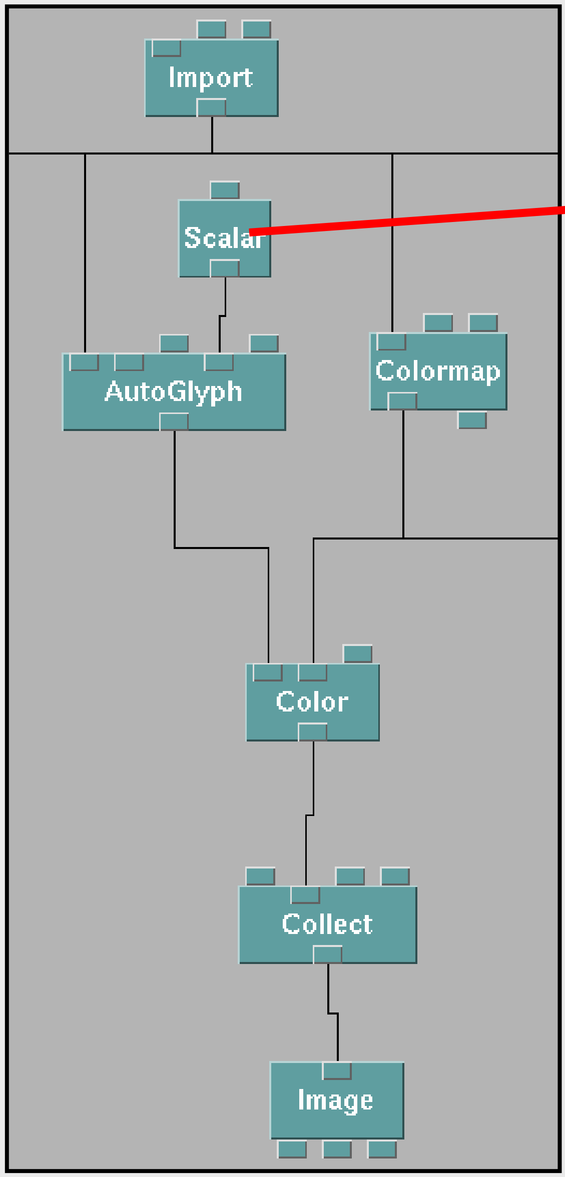
Specify the file to write to

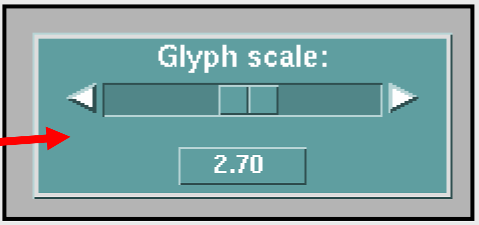


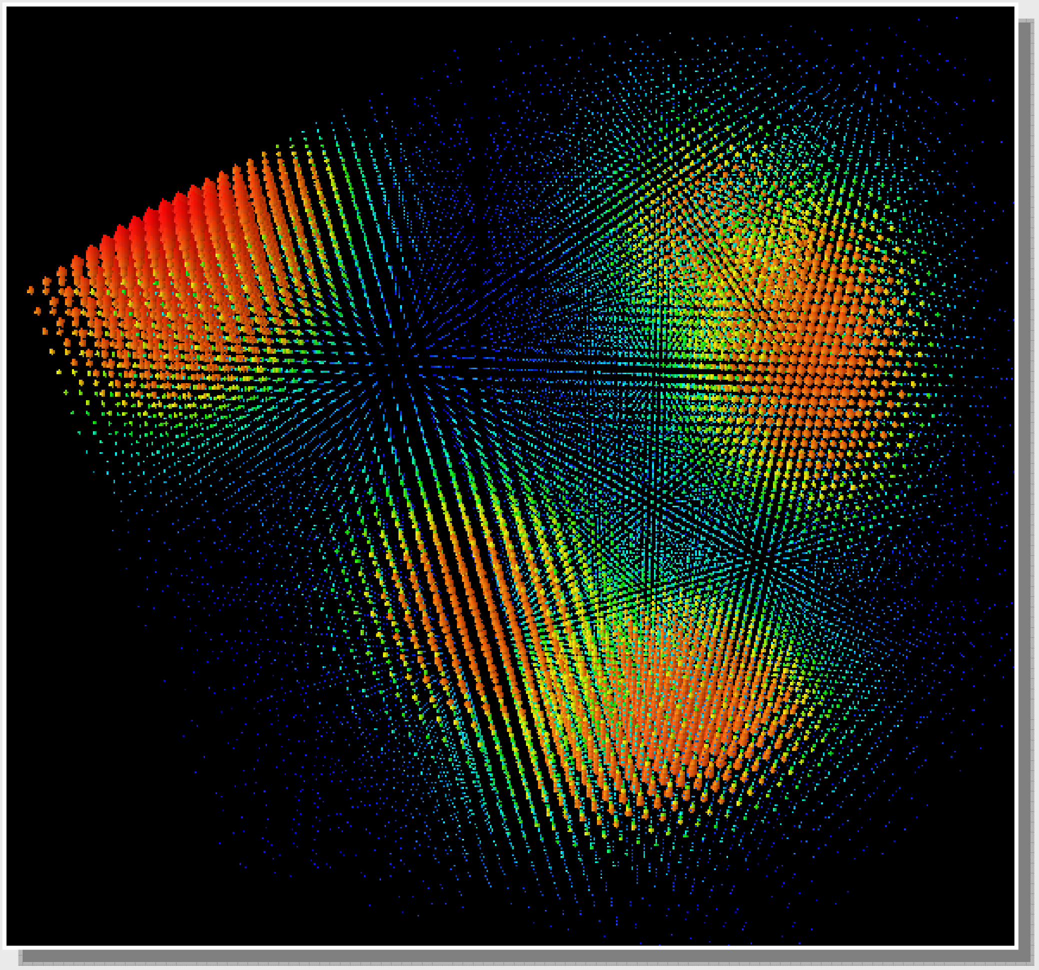
```
7        1.622        1.615        1.508
001         1.603        1.623        1.554
 Attribute.  Name 'dep':
  String.  "positions"
Component number 1, name 'positions':
 Product Array.  2 terms.
 Product term 0:   Regular Array.  201 items, float, real, 2-vector
  start value [ 0, 0 ], delta [ 1, 0 ], for 201 repetitions
 Product term 1:   Regular Array.  105 items, float, real, 2-vector
  start value [ 0, 0 ], delta [ 0, -1 ], for 105 repetitions
 Attribute.  Name 'dep':
  String.  "positions"
Component number 2, name 'connections':
 Mesh Array.  2 terms.
 Mesh offset: 0, 0
 Mesh term 0:   Path Array.  connects 201 items
 Mesh term 1:   Path Array.  connects 105 items
 Attribute.  Name 'element type':
  String.  "quads"
 Attribute.  Name 'dep':
  String.  "connections"
 Attribute.  Name 'ref':
  String.  "positions"
Component number 3, name 'box':
 Generic Array.  4 items, float, real, 2-vector
 data values:
             0           -104
             0              0
           200           -104
           200              0

 Attribute.  Name 'der':
  String.  "positions"
Attribute.  Name 'name':
 String.  "field0"
```

Scalar Visualization

Glyphs



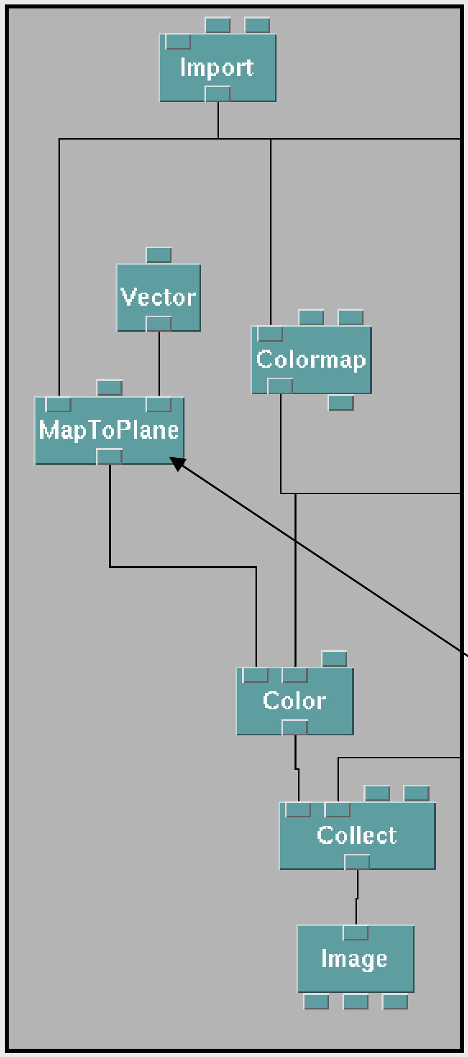




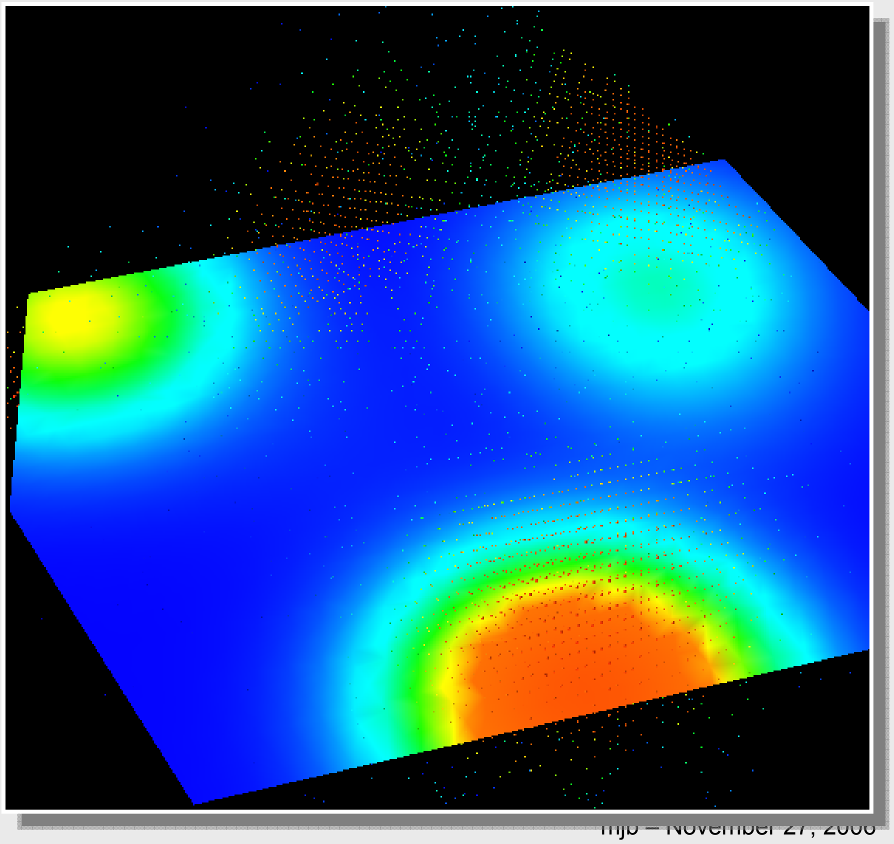
Oregon State University

mjb – November 27, 2006

3D Cutting Plane – Interpolated Colors

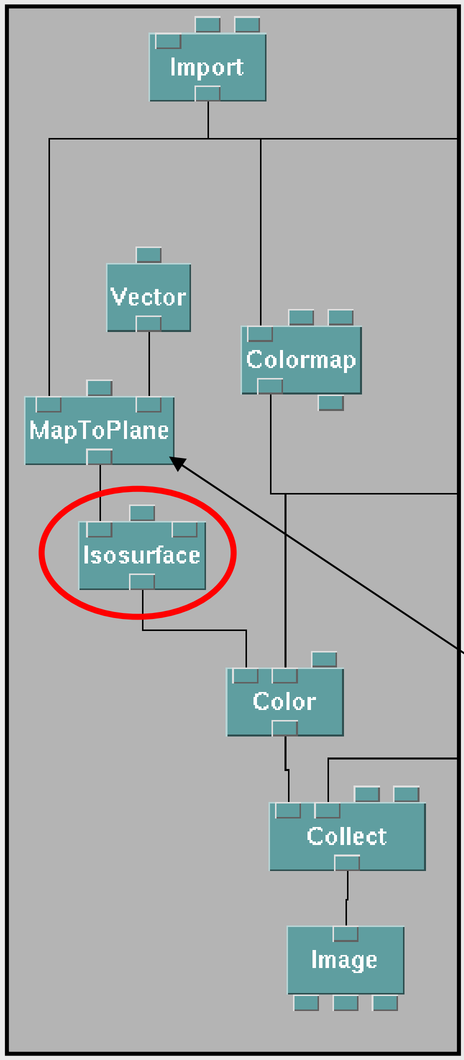


MapToPlane interpolates the 3D field onto the given plane. The first argument is the field, the second is a 3D point on the plane, and the third argument is a 3D normal to the plane.

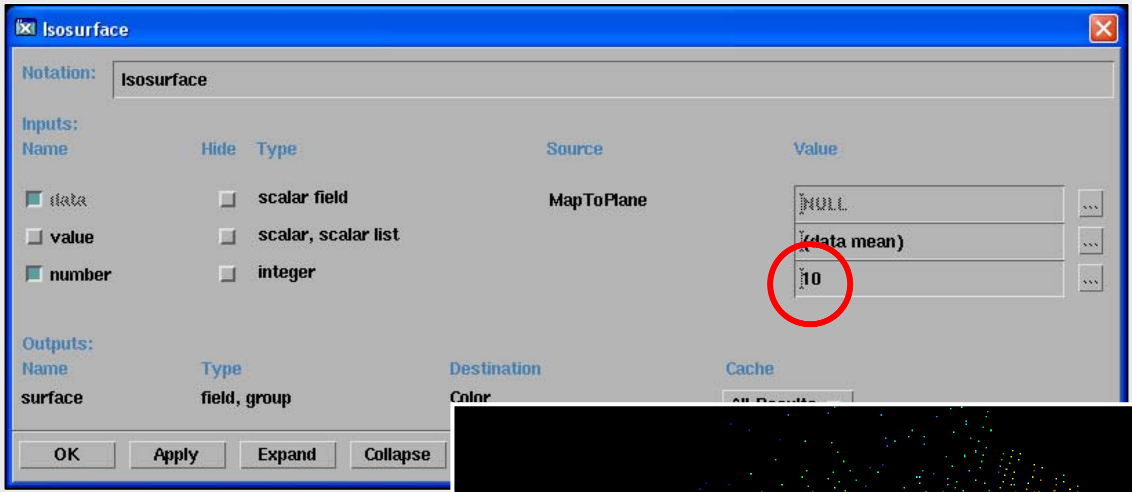


Oregon State University

3D Cutting Plane – Contours



Realization



Isosurface

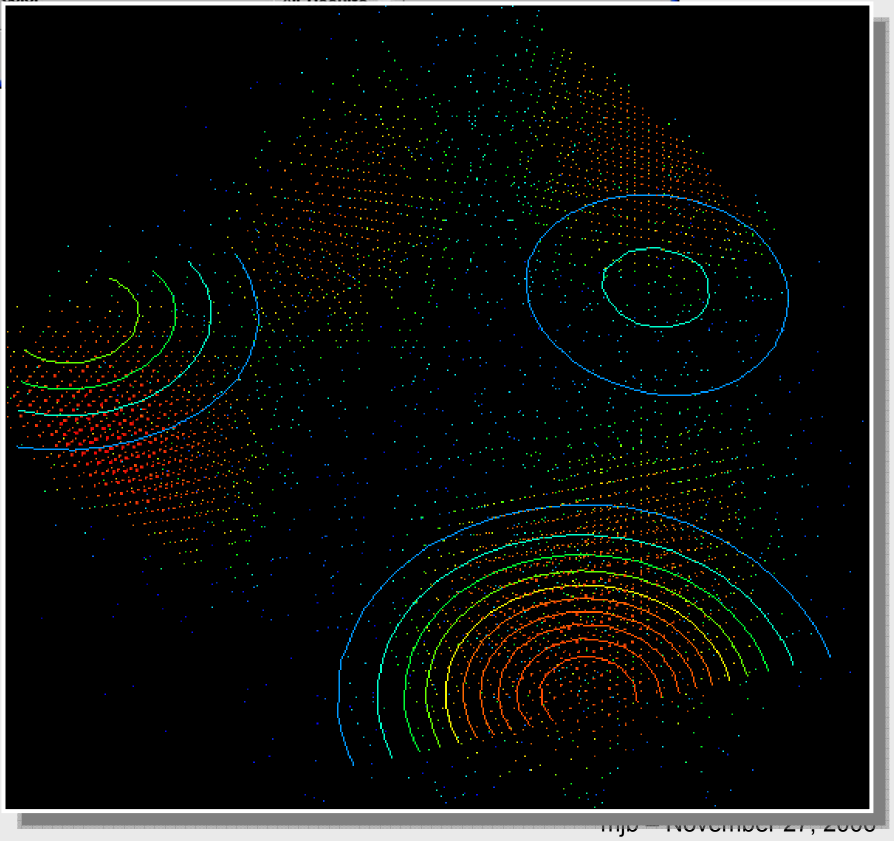
Notation: Isosurface

Inputs:

Name	Hide	Type	Source	Value
☑ data	☐	scalar field	MapToPlane	NULL
☐ value	☐	scalar, scalar list		(data mean)
☑ number	☐	integer		10

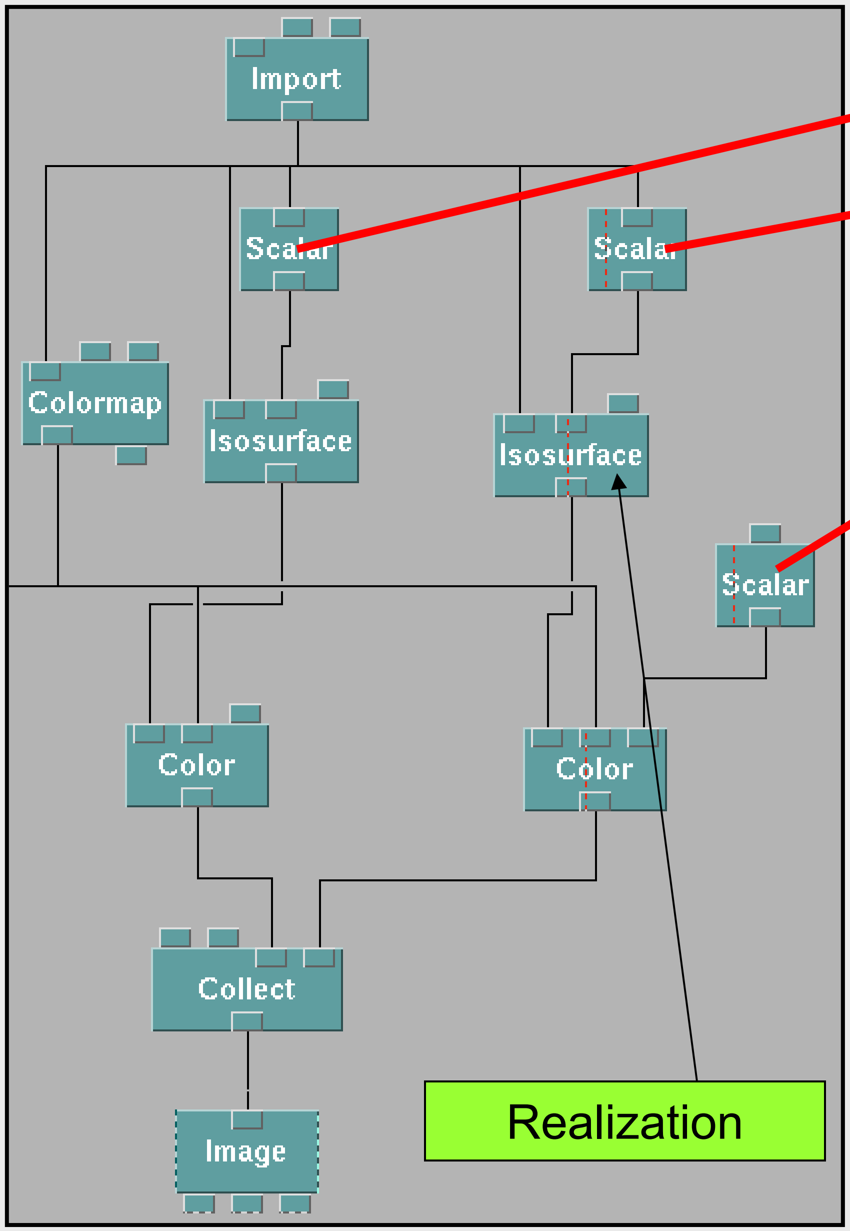
Outputs:

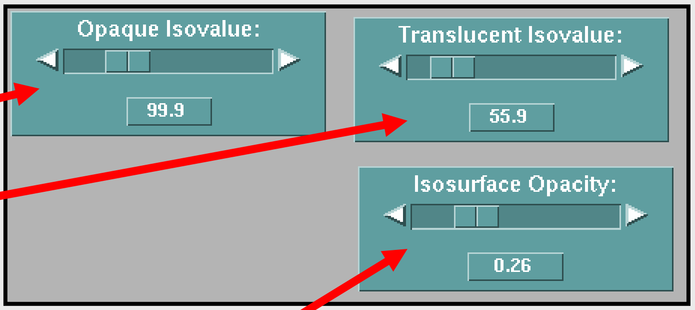
Name	Type	Destination	Cache
surface	field, group	Color	All Results



Oregon State University

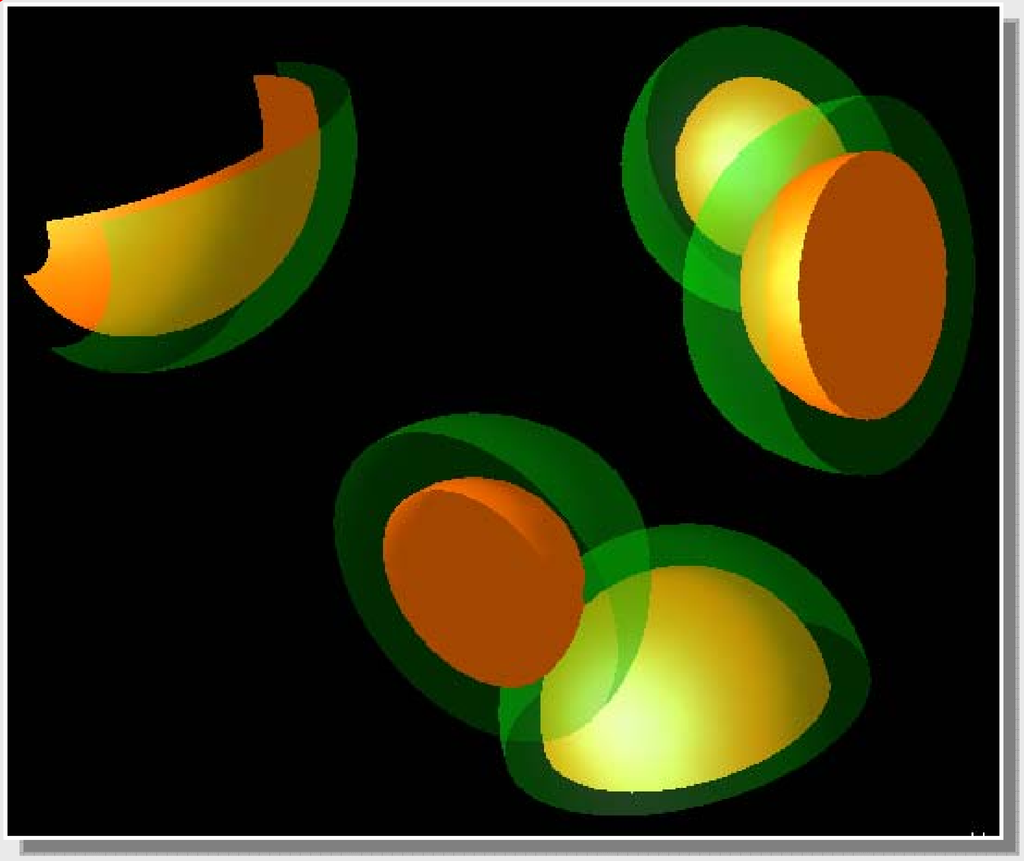
Isosurfaces




Opaque Isovalue: 99.9

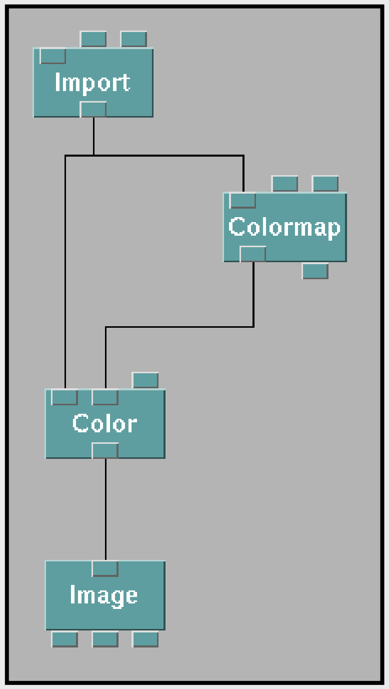
Translucent Isovalue: 55.9

Isosurface Opacity: 0.26



OSU Oregon State University

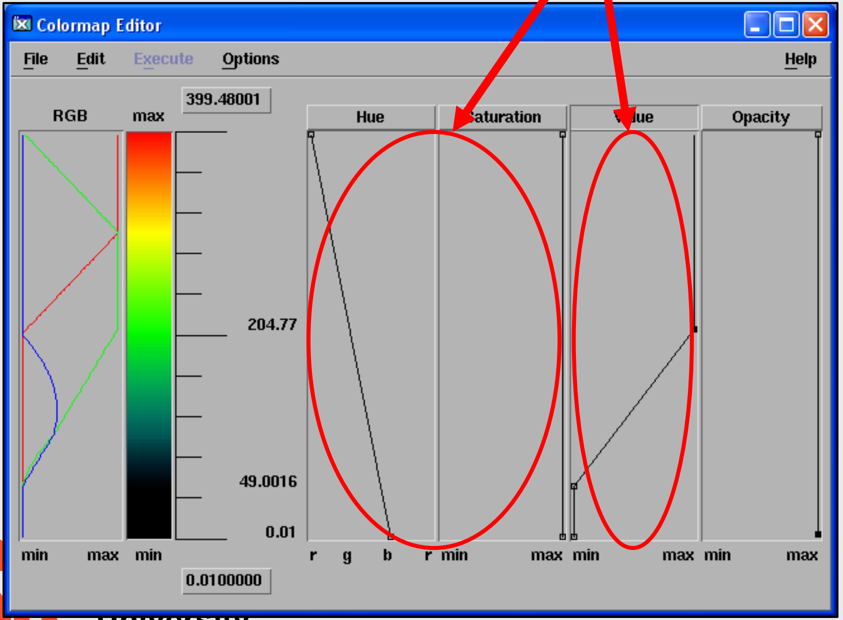
Direct Volume Rendering

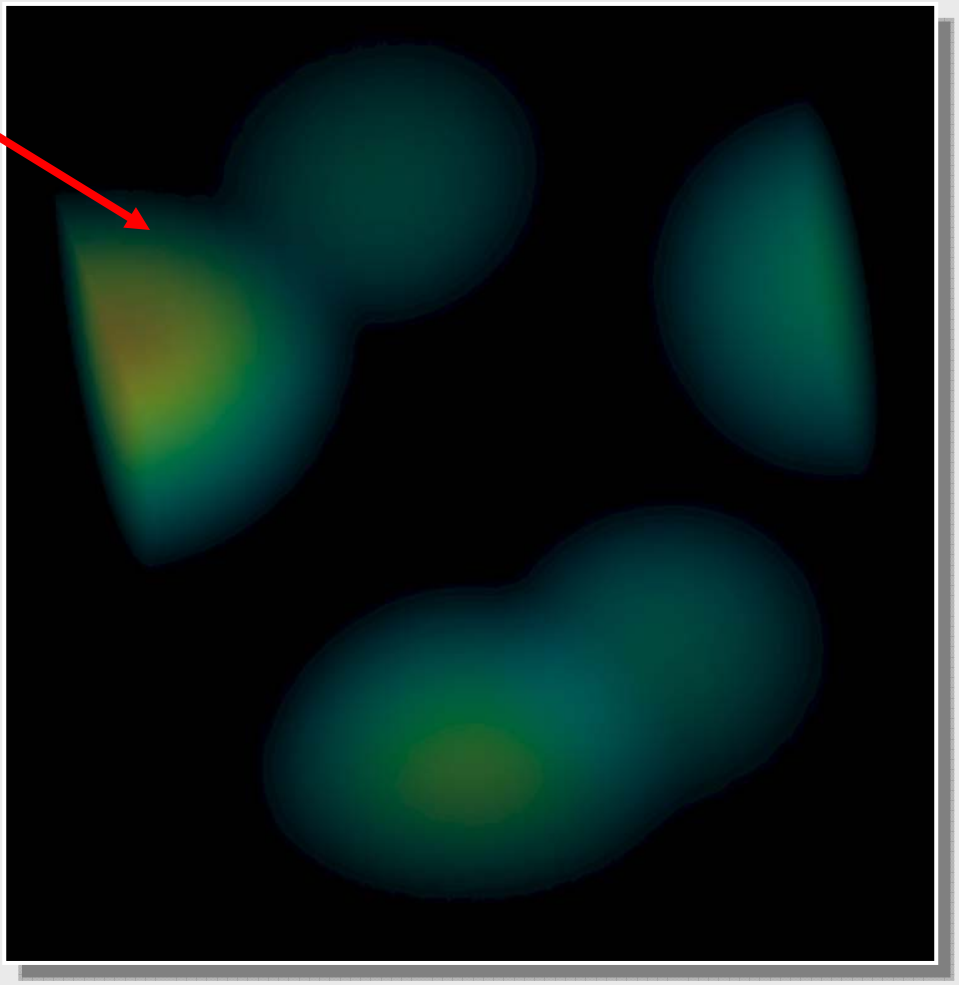


A Volume Rendering "Transfer Function" relates data scalar value to its corresponding color and opacity. For volume rendering, OpenDX uses the color *Value* as the opacity, not the color Opacity.

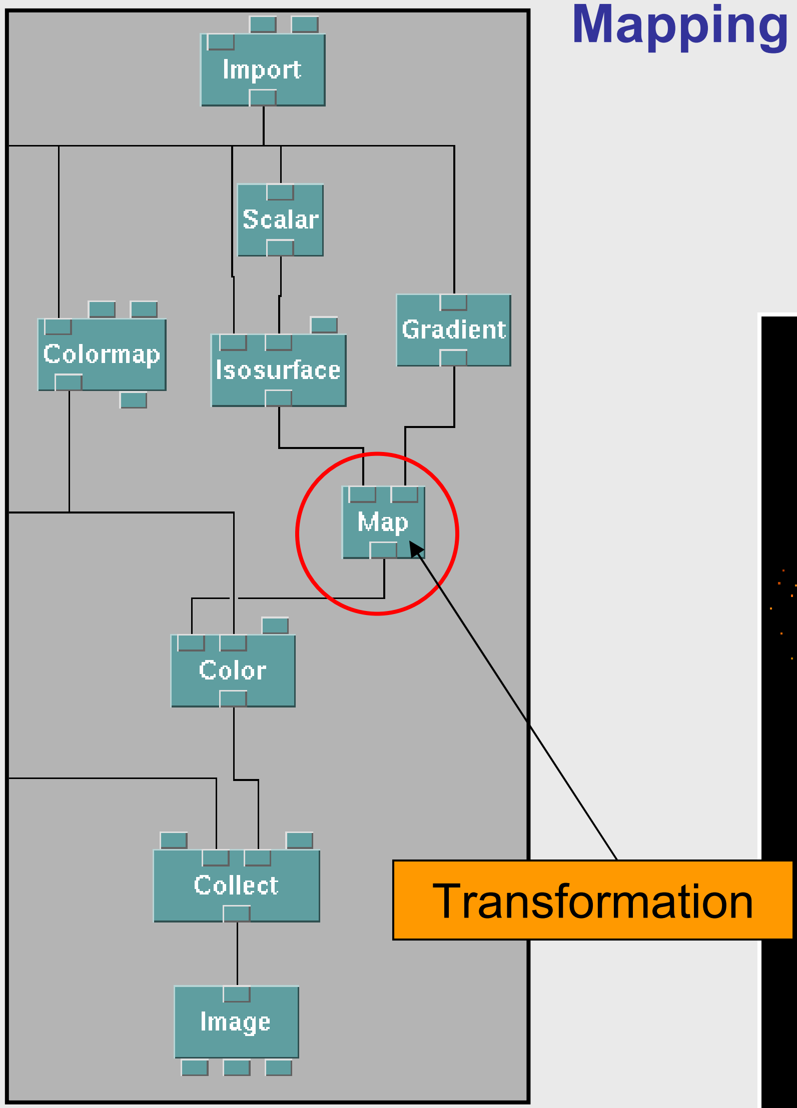
The direct volume rendering part of the Image module will only work in Orthographic projection.

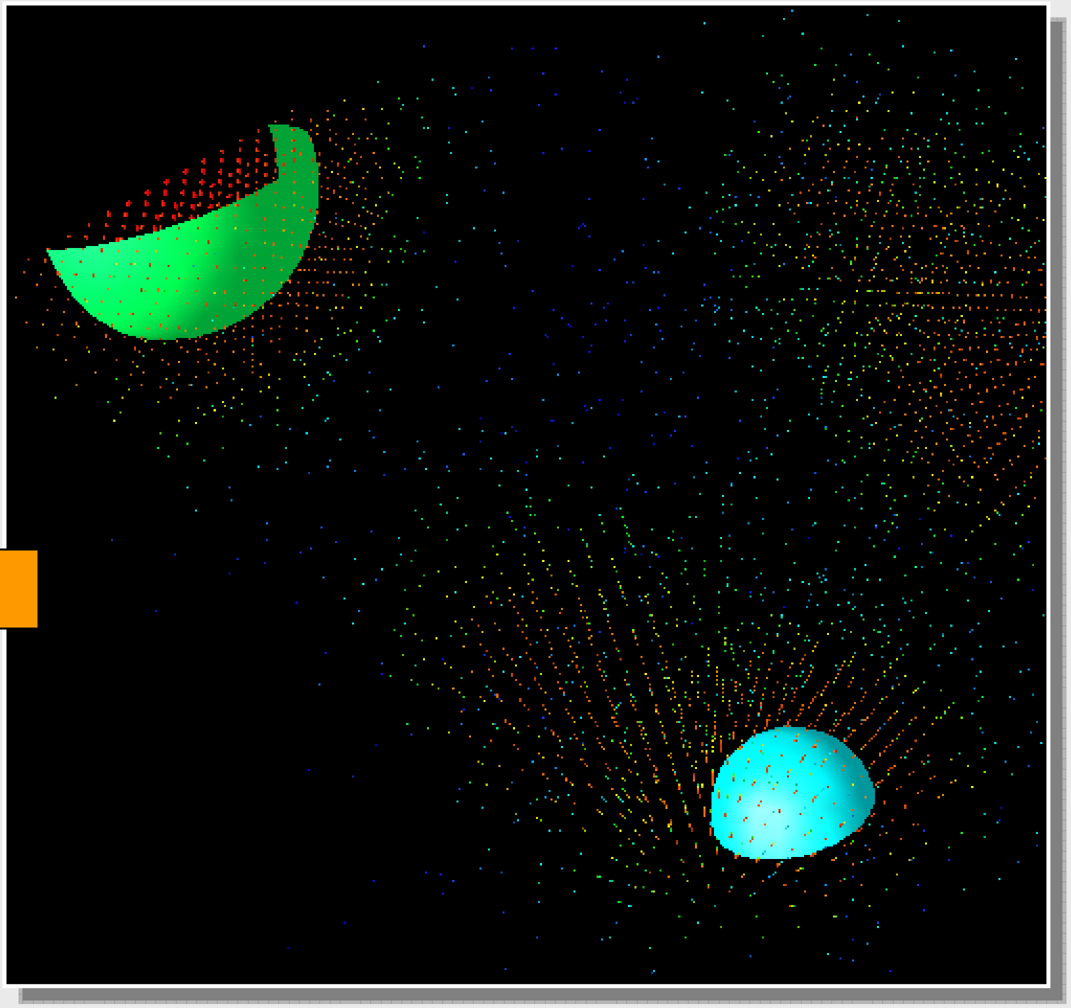
These are the "Transfer Function"





Mapping Another Data Field onto Isosurfaces

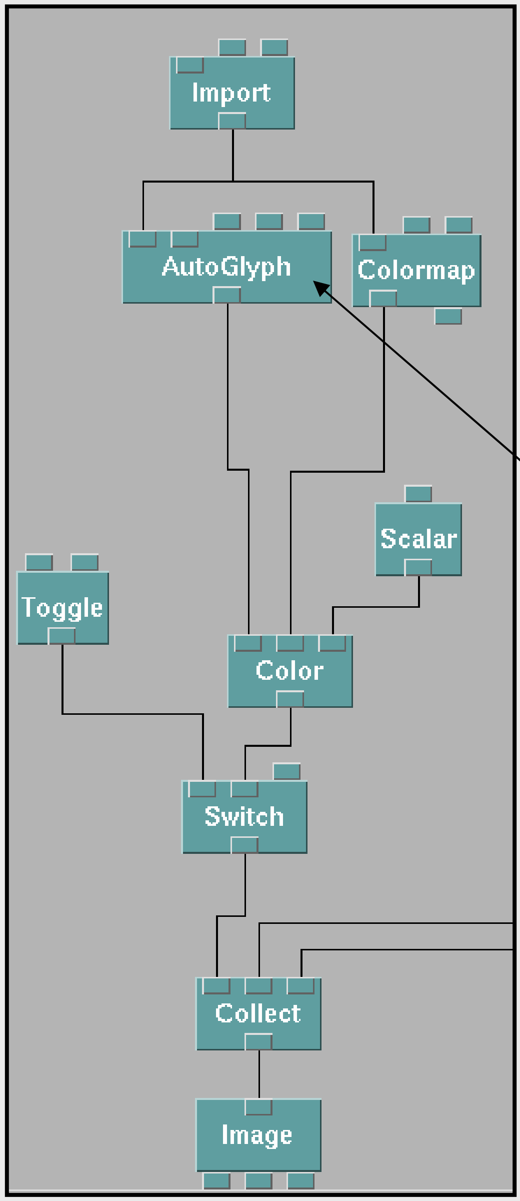




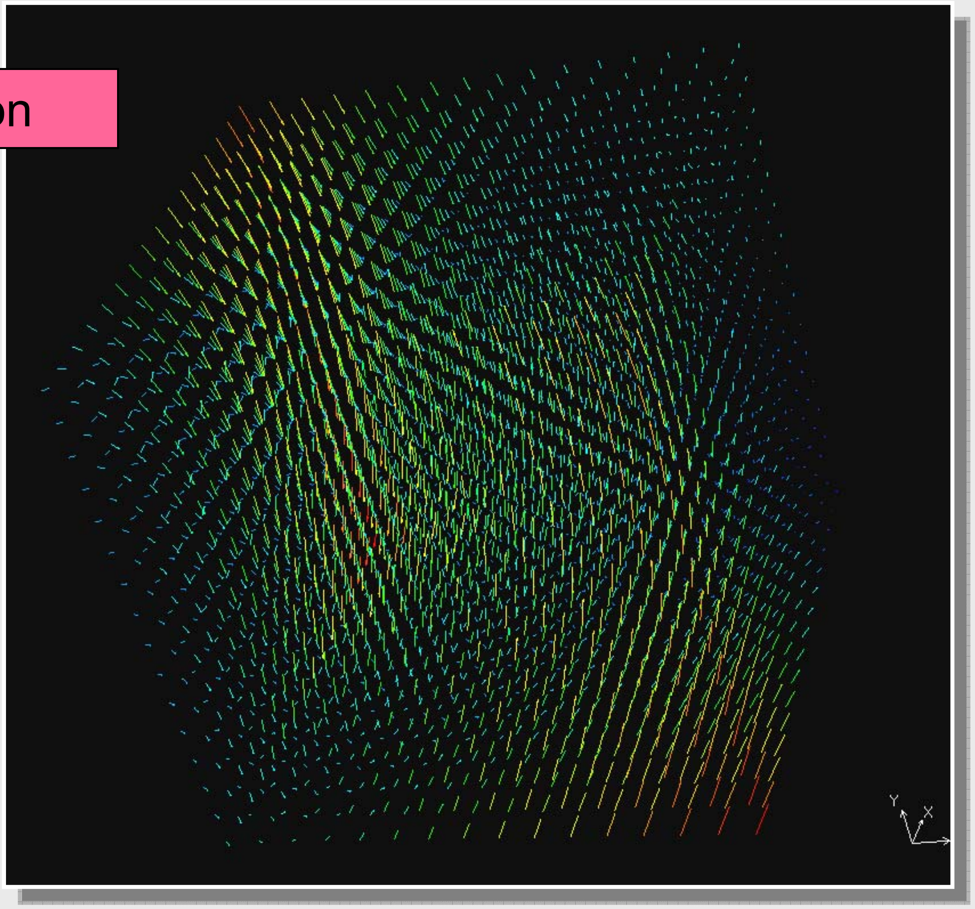
Oregon State University

Vector Visualization

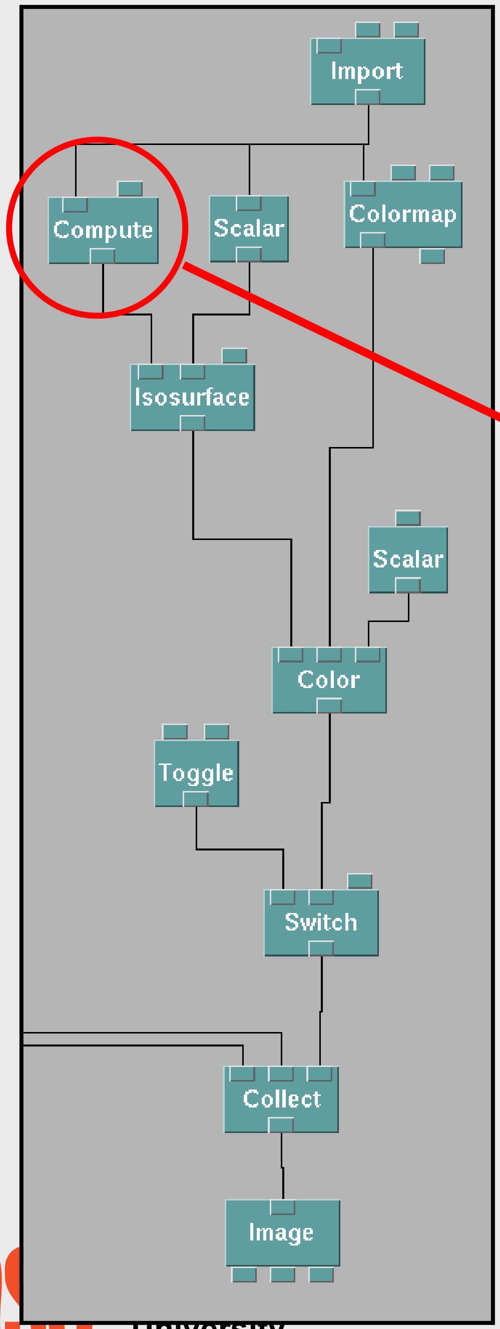
Vector Cloud

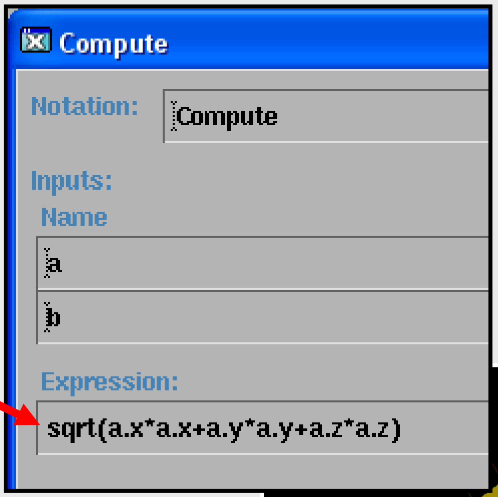


Annotation



Speed Isosurfaces




Compute

Notation: Compute

Inputs:

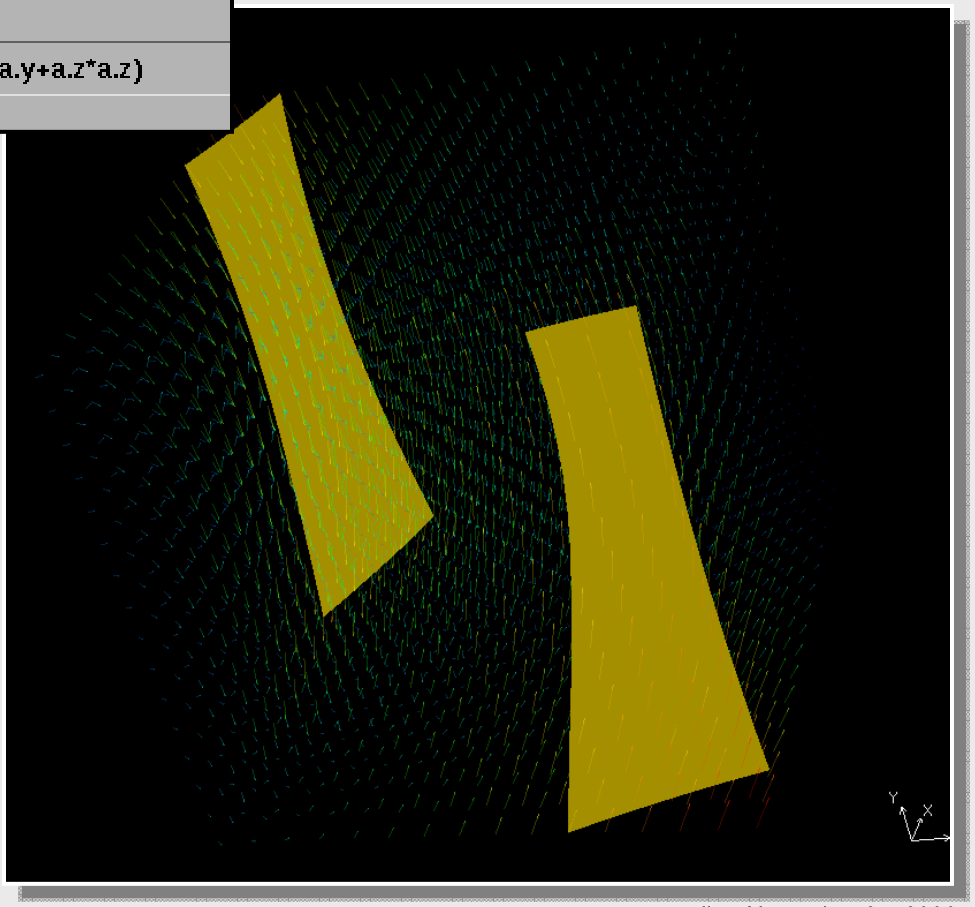
Name

a

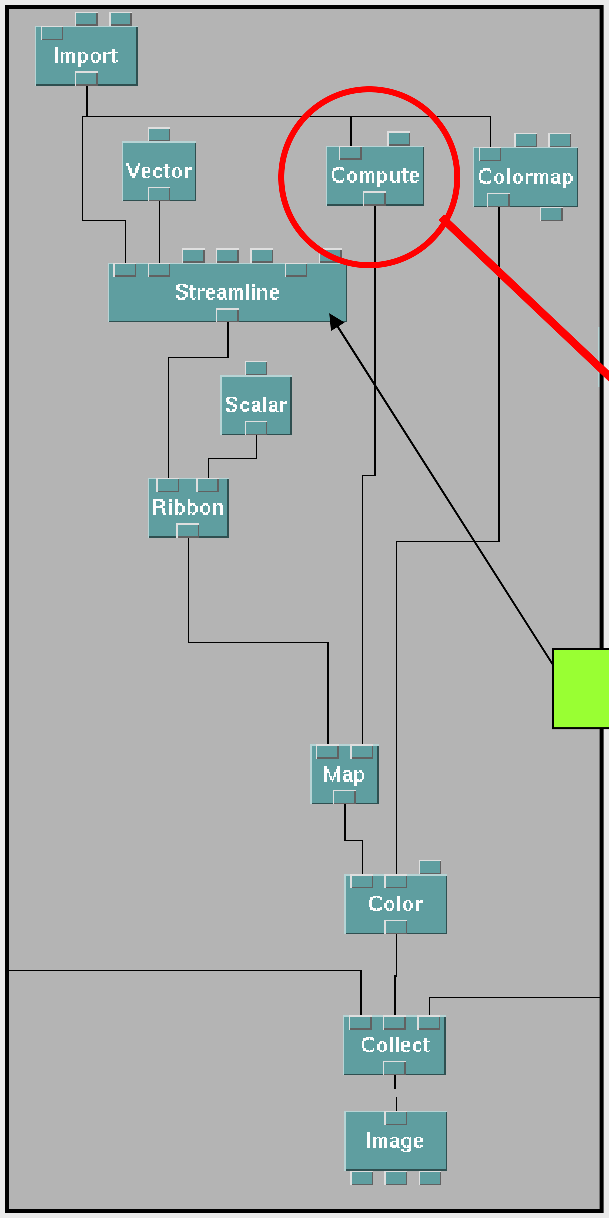
b

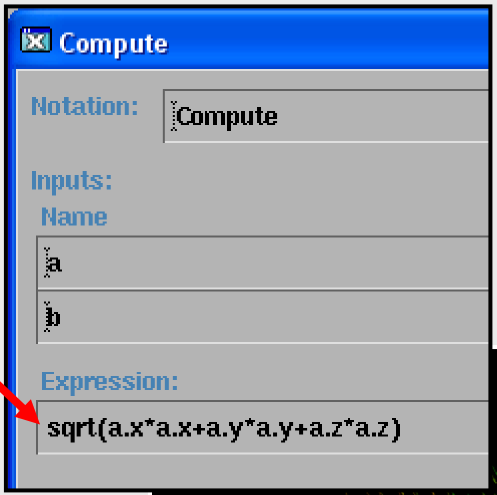
Expression:

sqrt(a.x*a.x+a.y*a.y+a.z*a.z)



Streamline Ribbon




Compute

Notation: Compute

Inputs:

Name

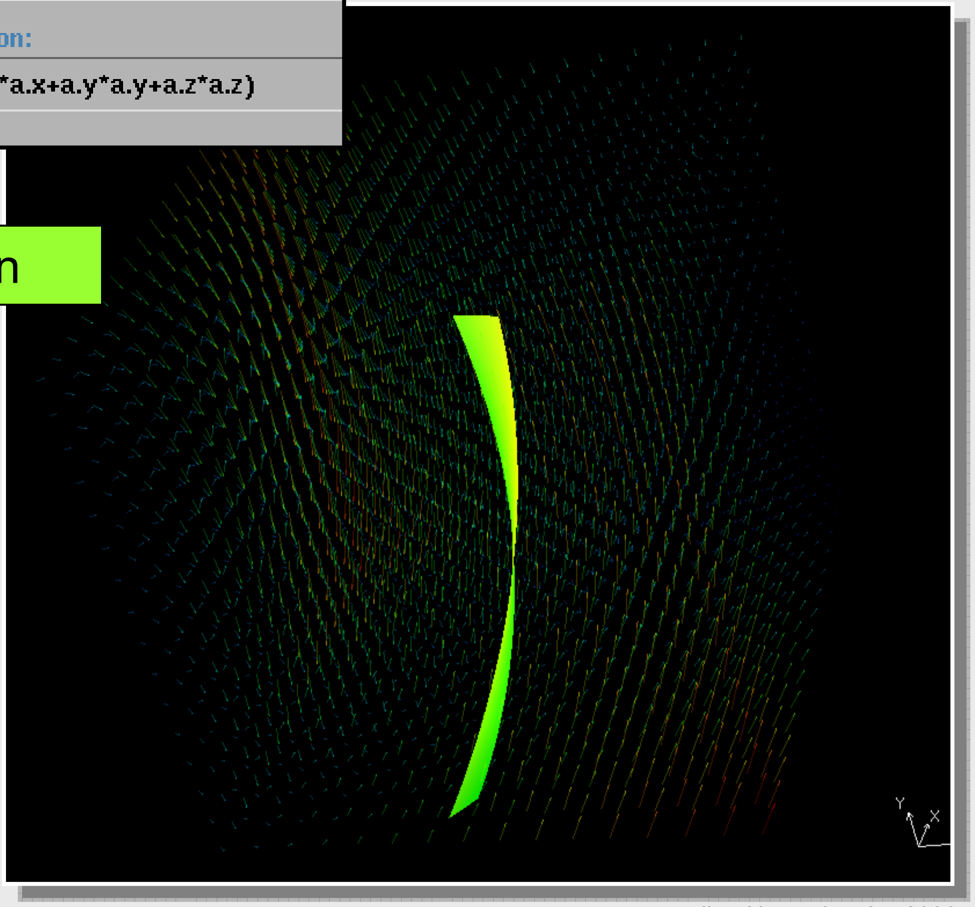
a

b

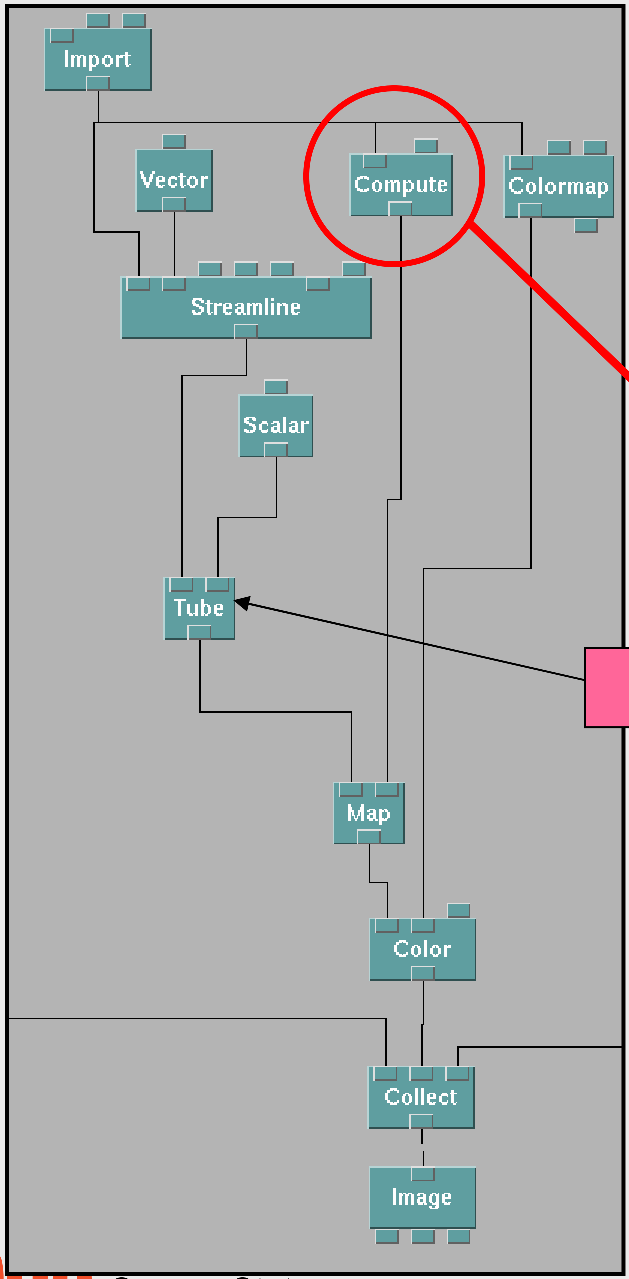
Expression:

sqrt(a.x*a.x+a.y*a.y+a.z*a.z)

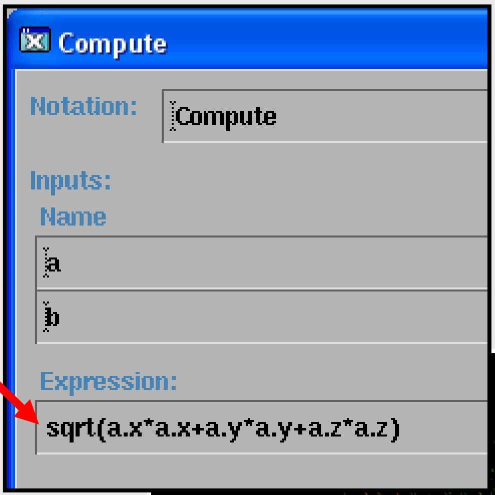
Realization



Streamline Tube



Annotation


Compute

Notation: Compute

Inputs:

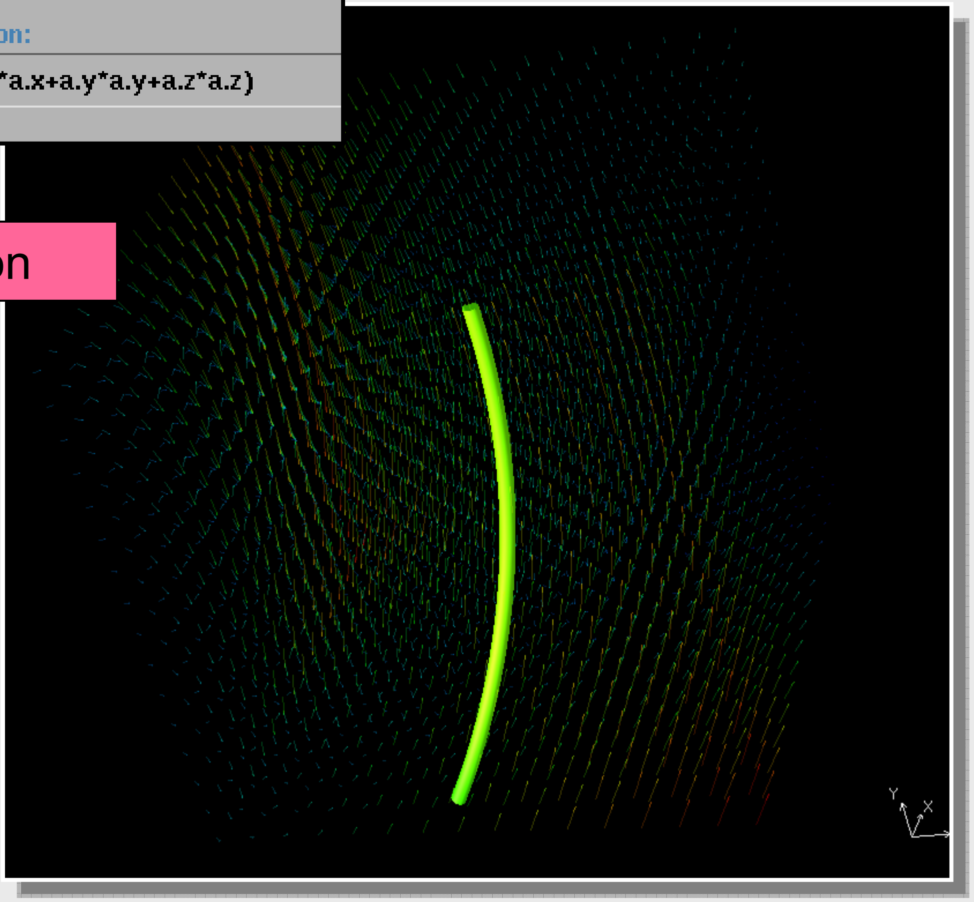
Name

a

b

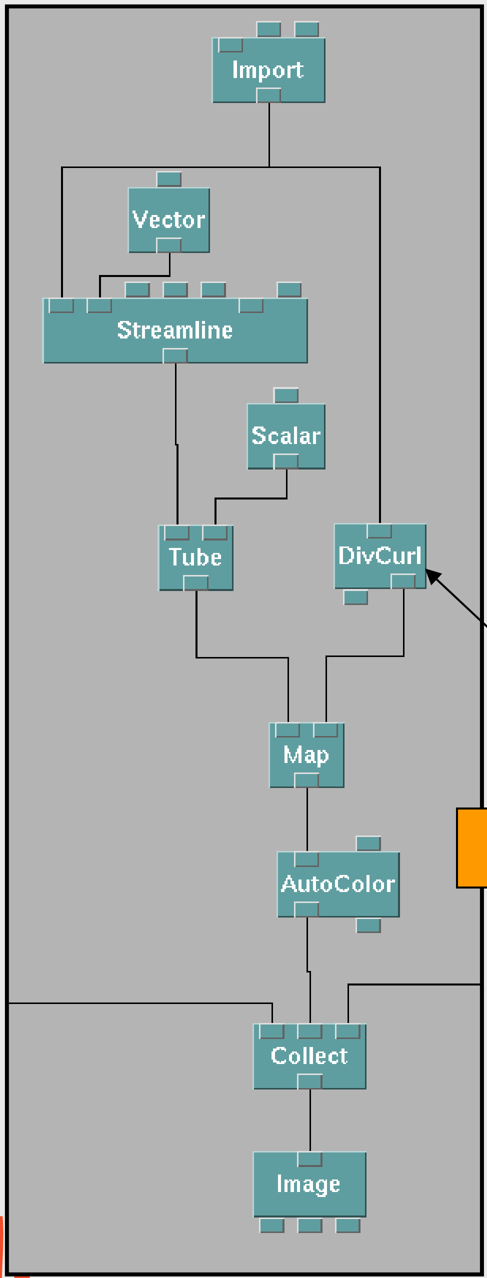
Expression:

```
sqrt(a.x*a.x+a.y*a.y+a.z*a.z)
```

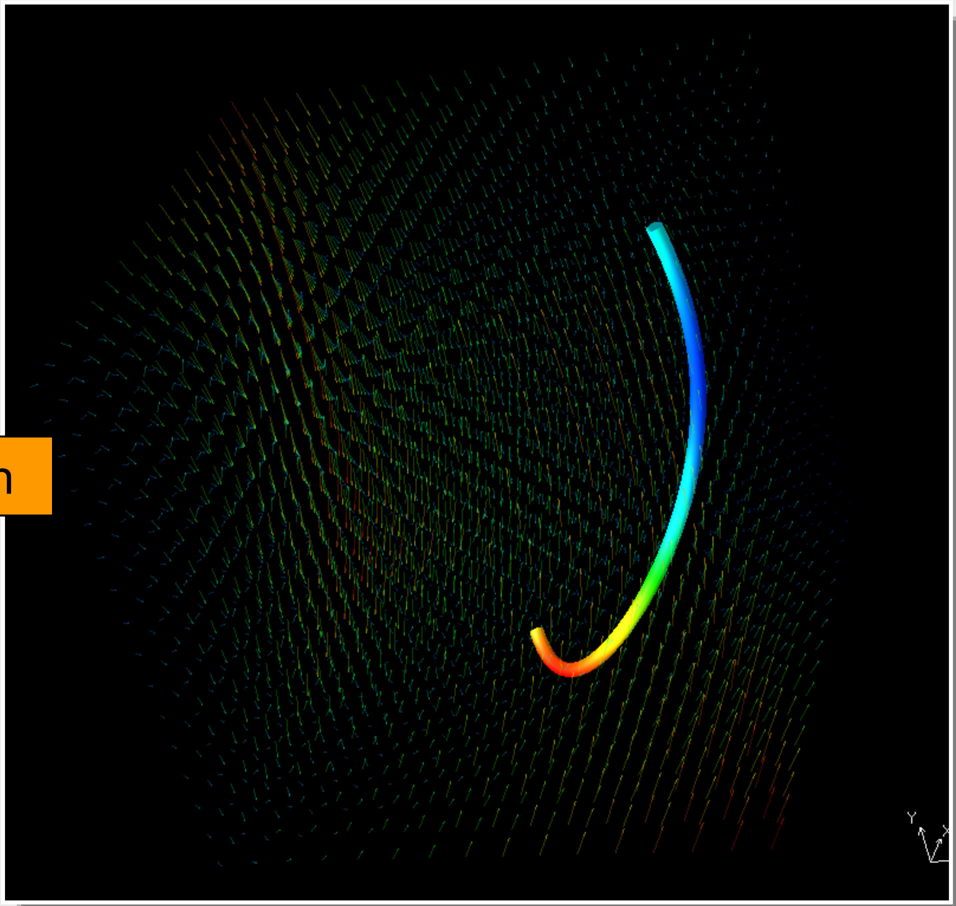


Oregon State University

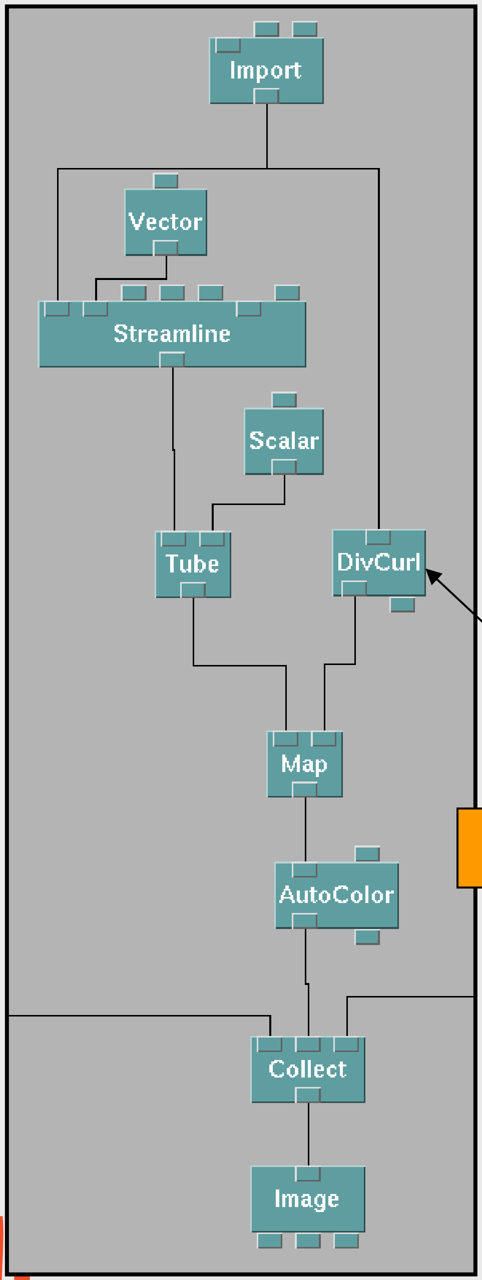
Curl



Transformation



Divergence

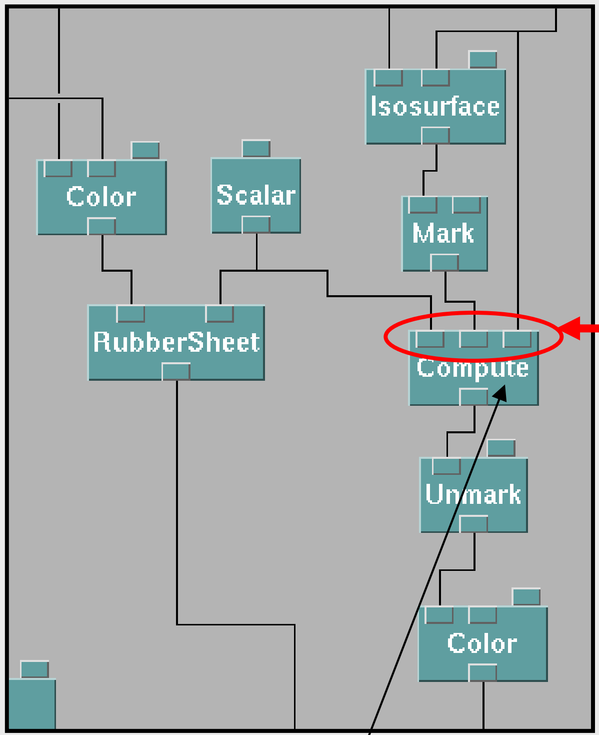


Transformation

Animation

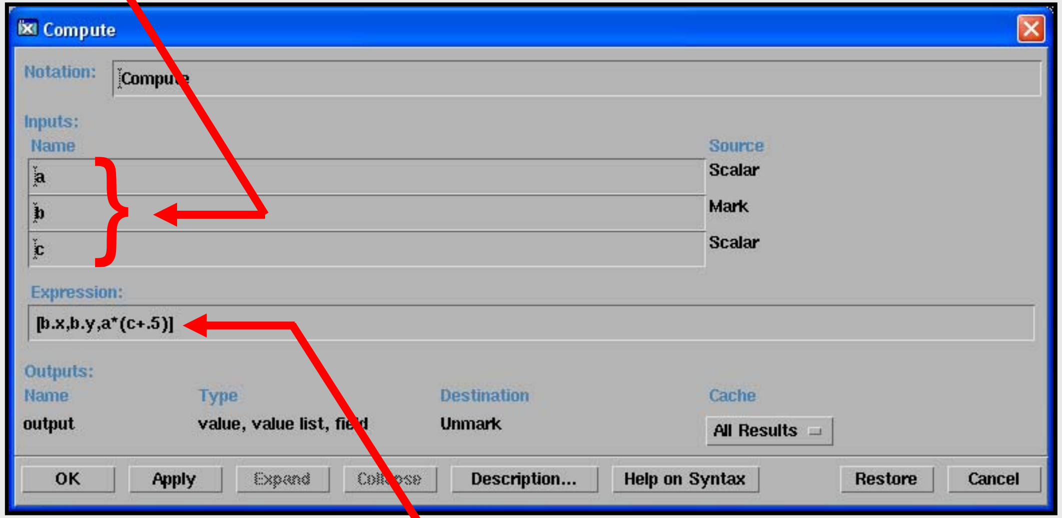
The Compute Module

Does arithmetic on the point-by-point Data component of a field, and outputs the modified field



Transformation

The 3 (in this case) inputs



Compute

Notation: Compute

Inputs:

Name	Source
a	Scalar
b	Mark
c	Scalar

Expression:

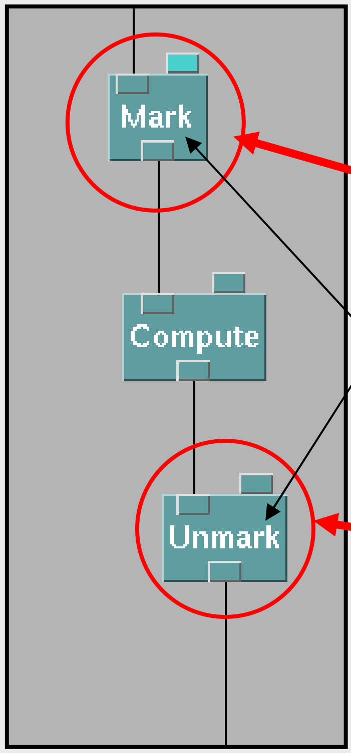
[b.x,b.y,a*(c+.5)]

Outputs:

Name	Type	Destination	Cache
output	value, value list, field	Unmark	All Results

The output expression, in this case, a 3-vector with a newly-created Z value

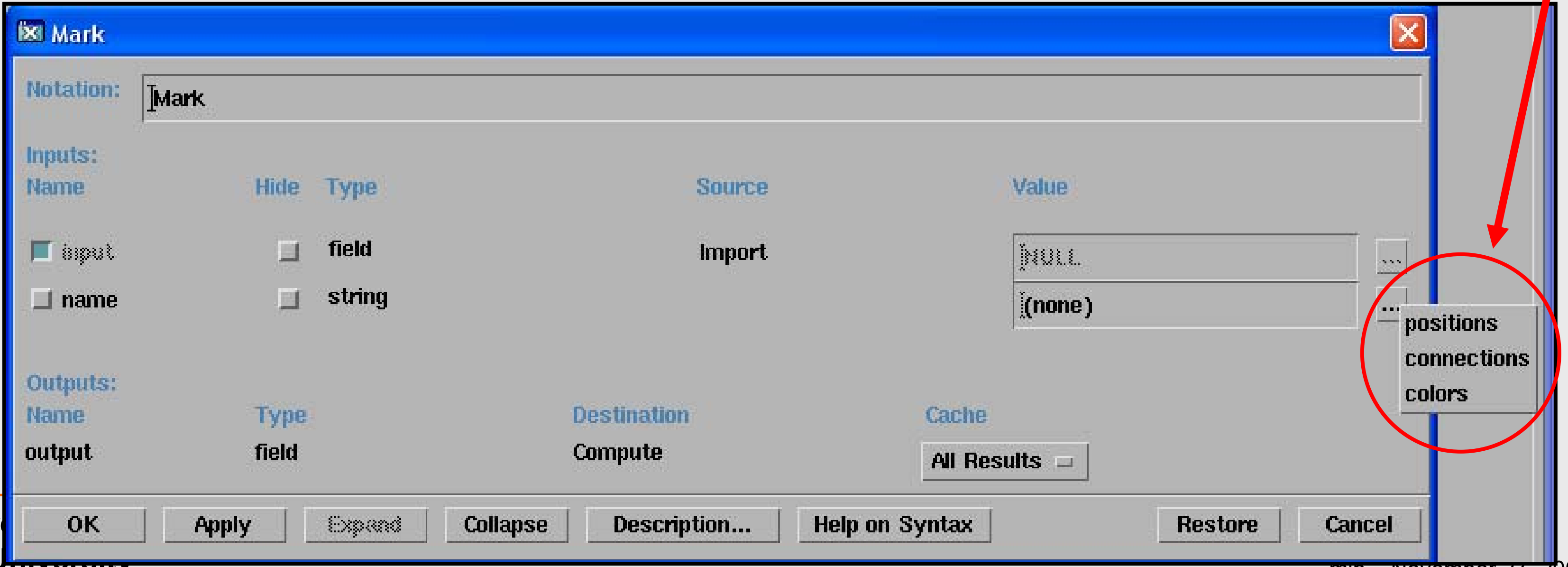
The Compute Module



Does arithmetic on the point-by-point Data component of a field, and outputs the modified field. But, what if you want to do arithmetic on a different component?

The *Mark* module renames the Data component to something temporary, and renames a component you select to "Data". *Compute* then acts on this component.

The *Unmark* module changes the component names back to what they were originally.



Mark

Notation: Mark

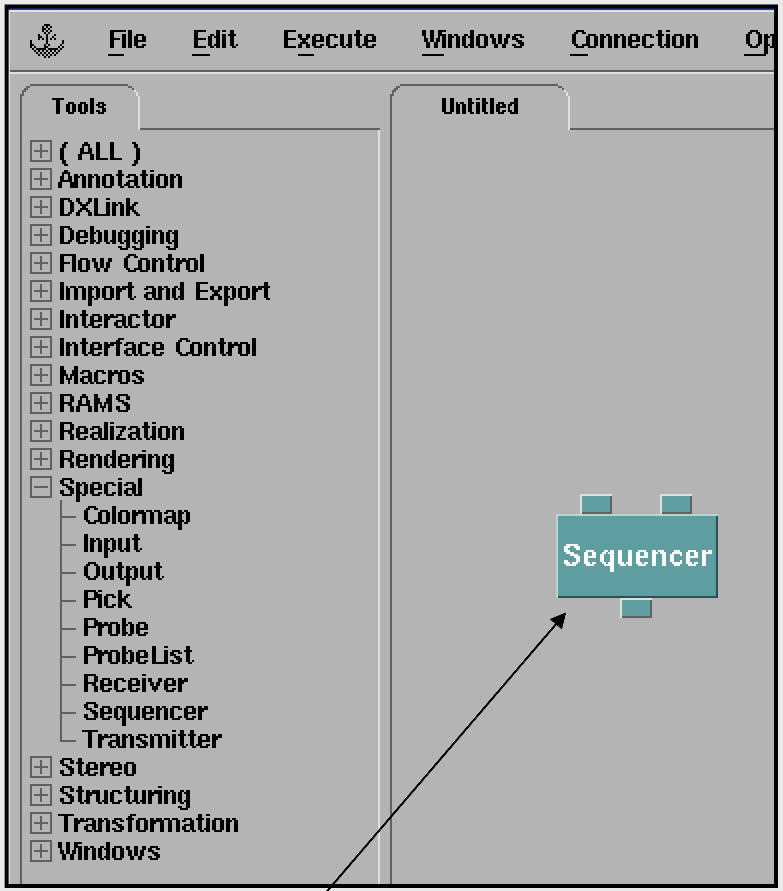
Inputs:

Name	Hide	Type	Source	Value
input		field	Import	NULL
name		string		(none)

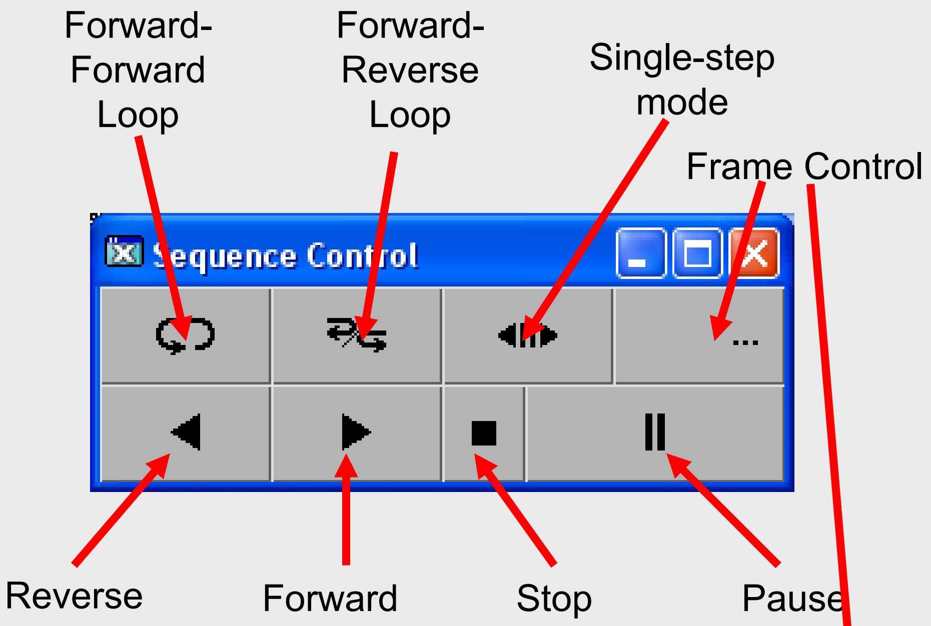
Outputs:

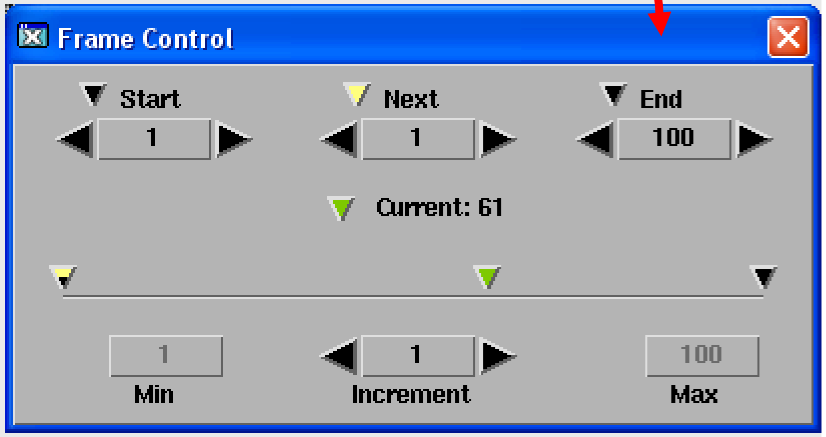
Name	Type	Destination	Cache
output	field	Compute	All Results

Animation: The Sequencer Module



Special

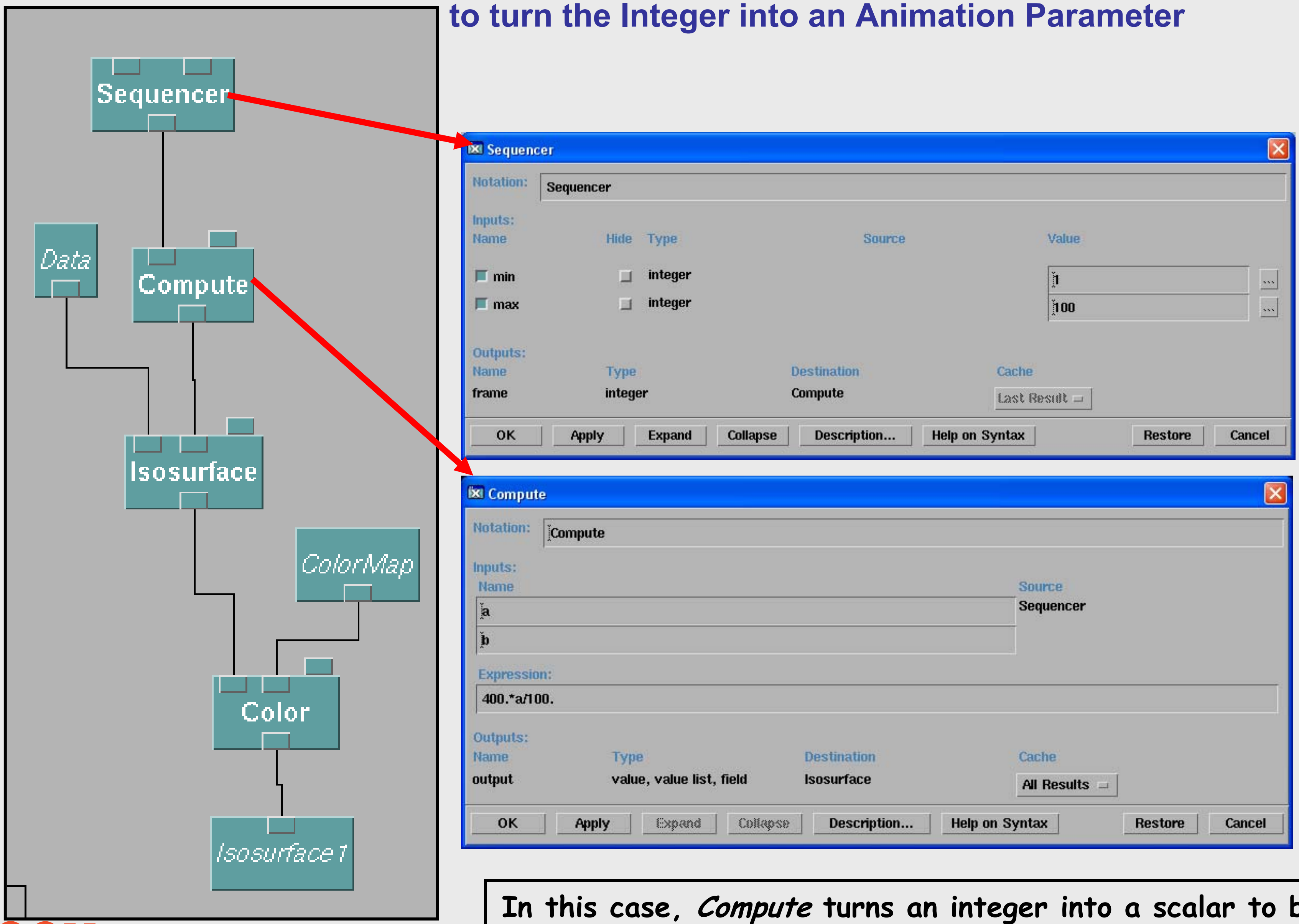




Sequencer outputs a series of integers. You set minimum, maximum, and delta using *Edit*→*Configuration*.

Oregon State University

The Sequencer Module: Usually Used with the Compute Module to turn the Integer into an Animation Parameter



Sequencer

Notation: Sequencer

Inputs:

Name	Hide	Type	Source	Value
min		integer		1
max		integer		100

Outputs:

Name	Type	Destination	Cache
frame	integer	Compute	Last Result

Compute

Notation: Compute

Inputs:

Name	Source
a	Sequencer
b	

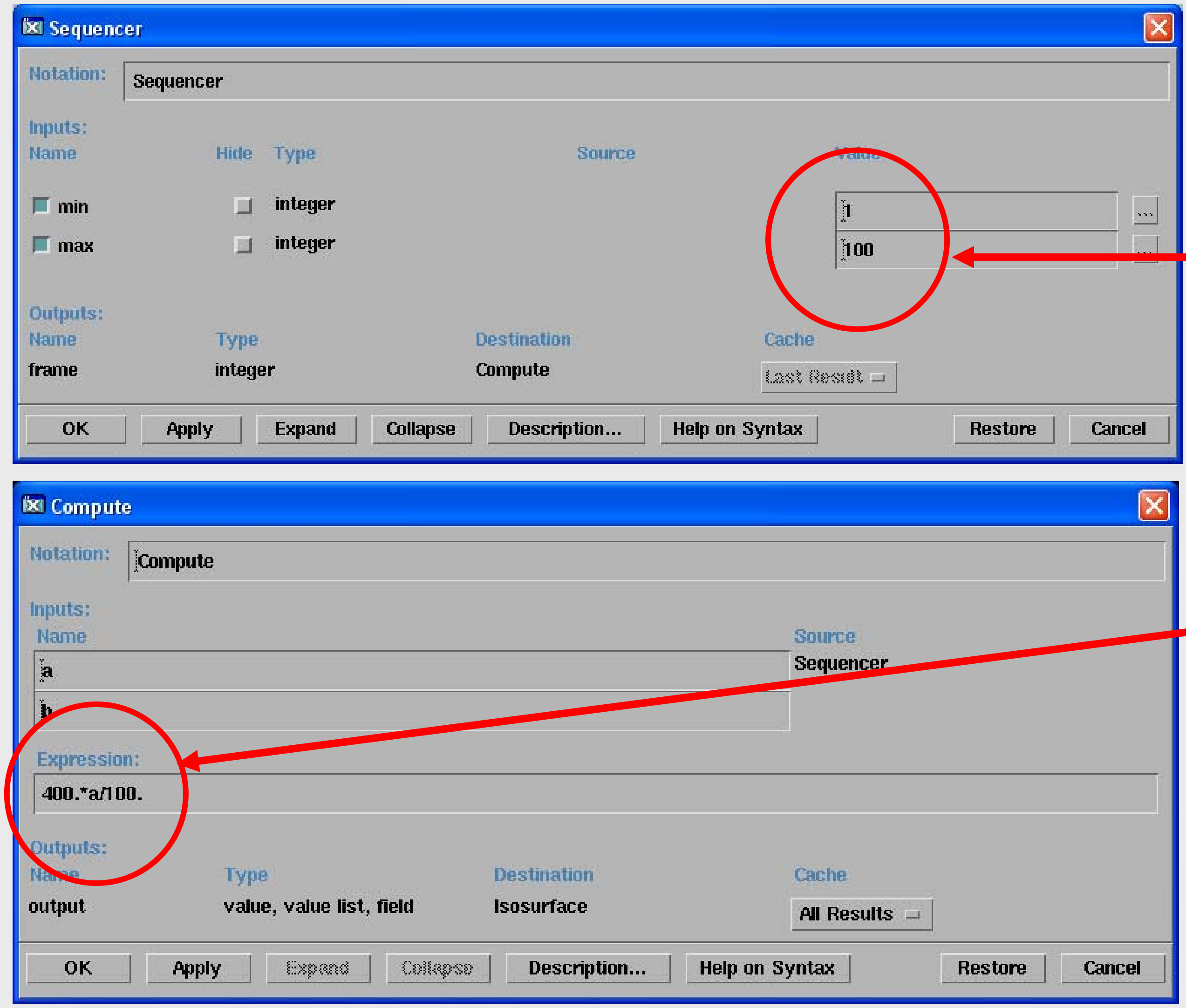
Expression: 400.*a/100.

Outputs:

Name	Type	Destination	Cache
output	value, value list, field	Isosurface	All Results

In this case, *Compute* turns an integer into a scalar to be used to animate an isovalue

The Sequencer Module: “Percent Units Strategy”



Notation: Sequencer

Inputs:

Name	Hide	Type	Source	Value
min		integer		1
max		integer		100

Outputs:

Name	Type	Destination	Cache
frame	integer	Compute	Last Result

Notation: Compute

Inputs:

Name	Source
a	Sequencer
b	

Expression: 400.*a/100.

Outputs:

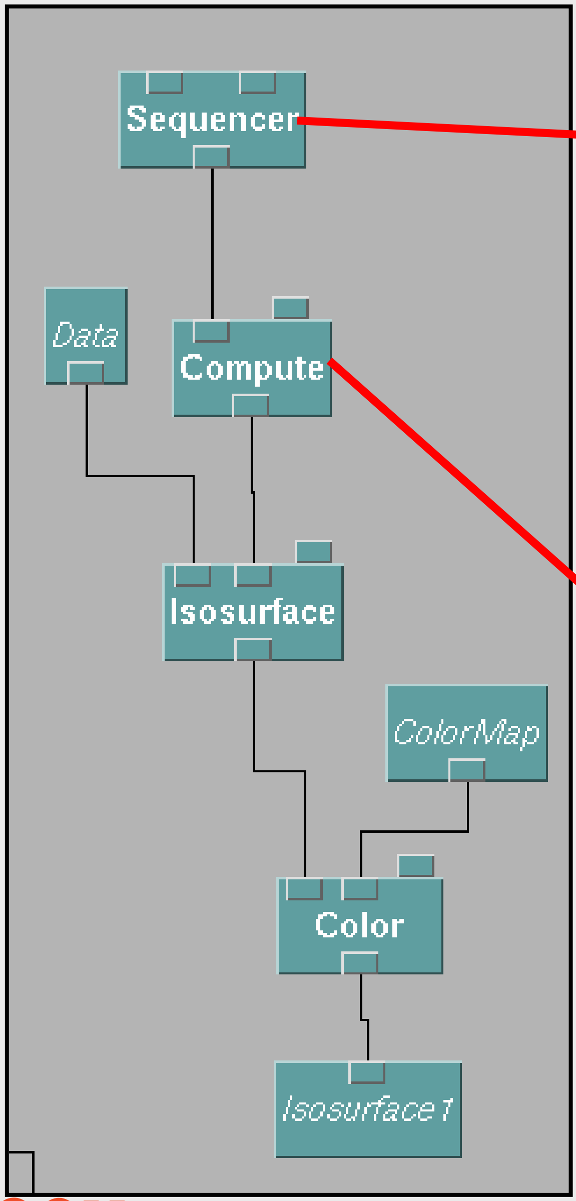
Name	Type	Destination	Cache
output	value, value list, field	Isosurface	All Results

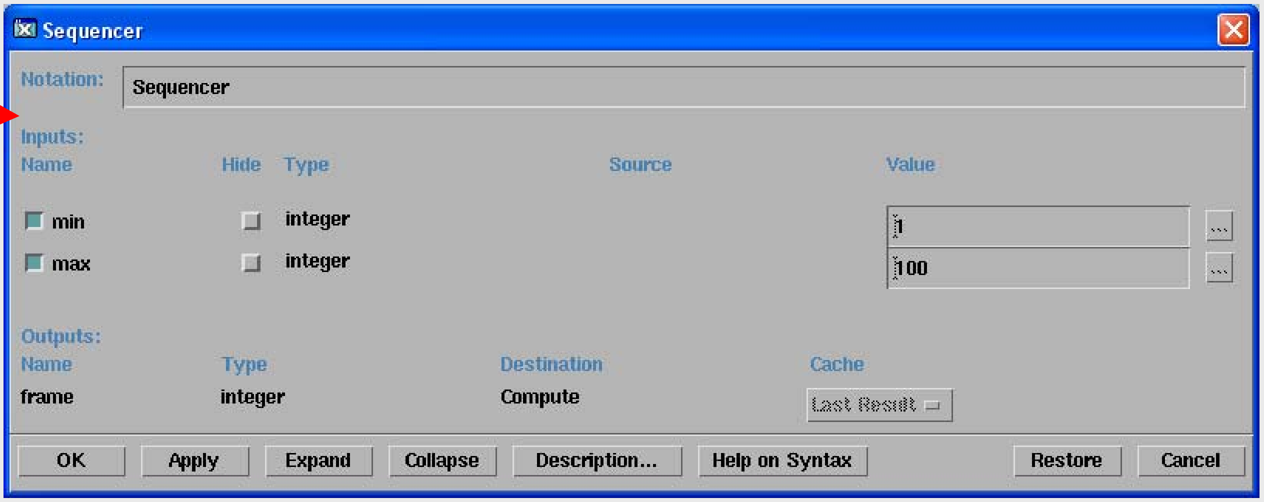
A good Sequencer Strategy: Run the sequence from 1-100 (or 0-100).

Then, base the *Compute* quantity on these “Percent Units”.

Oregon State University

The Sequencer Module: Setting a Scalar Isovalue





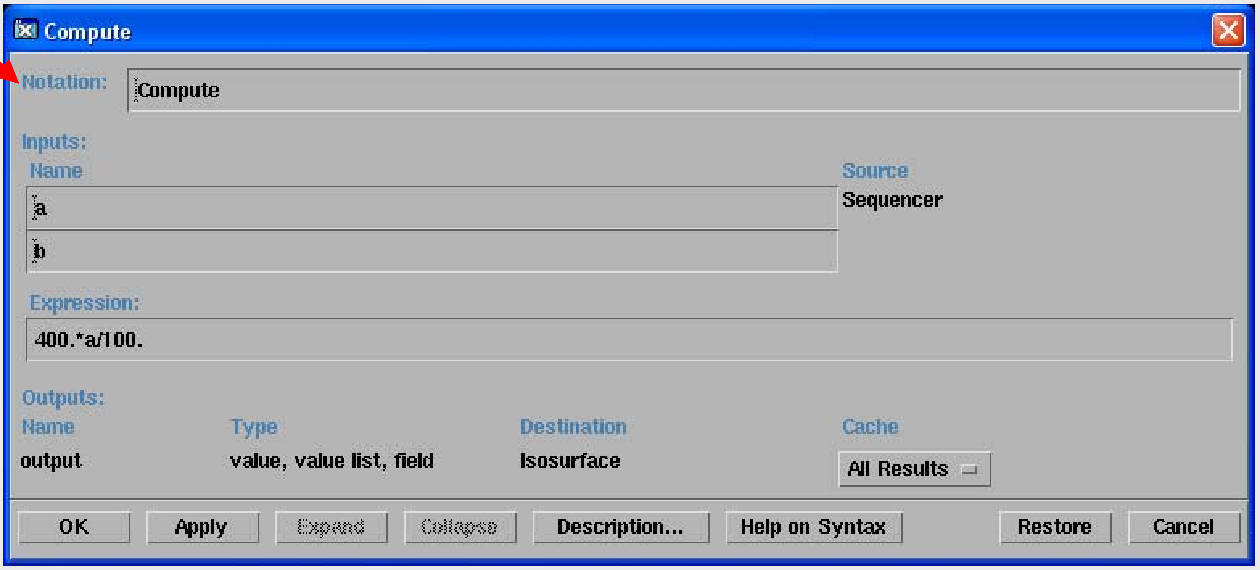
Notation: Sequencer

Inputs:

Name	Hide	Type	Source	Value
min		integer		1
max		integer		100

Outputs:

Name	Type	Destination	Cache
frame	integer	Compute	Last Result



Notation: Compute

Inputs:

Name	Source
a	Sequencer
b	

Expression:

400.*a/100.

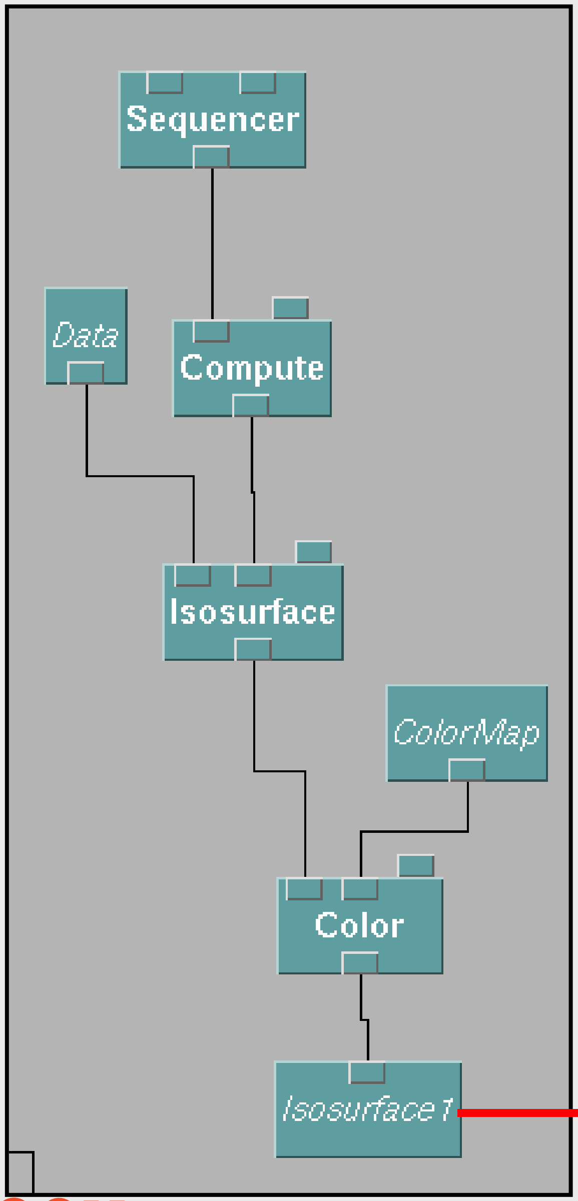
Outputs:

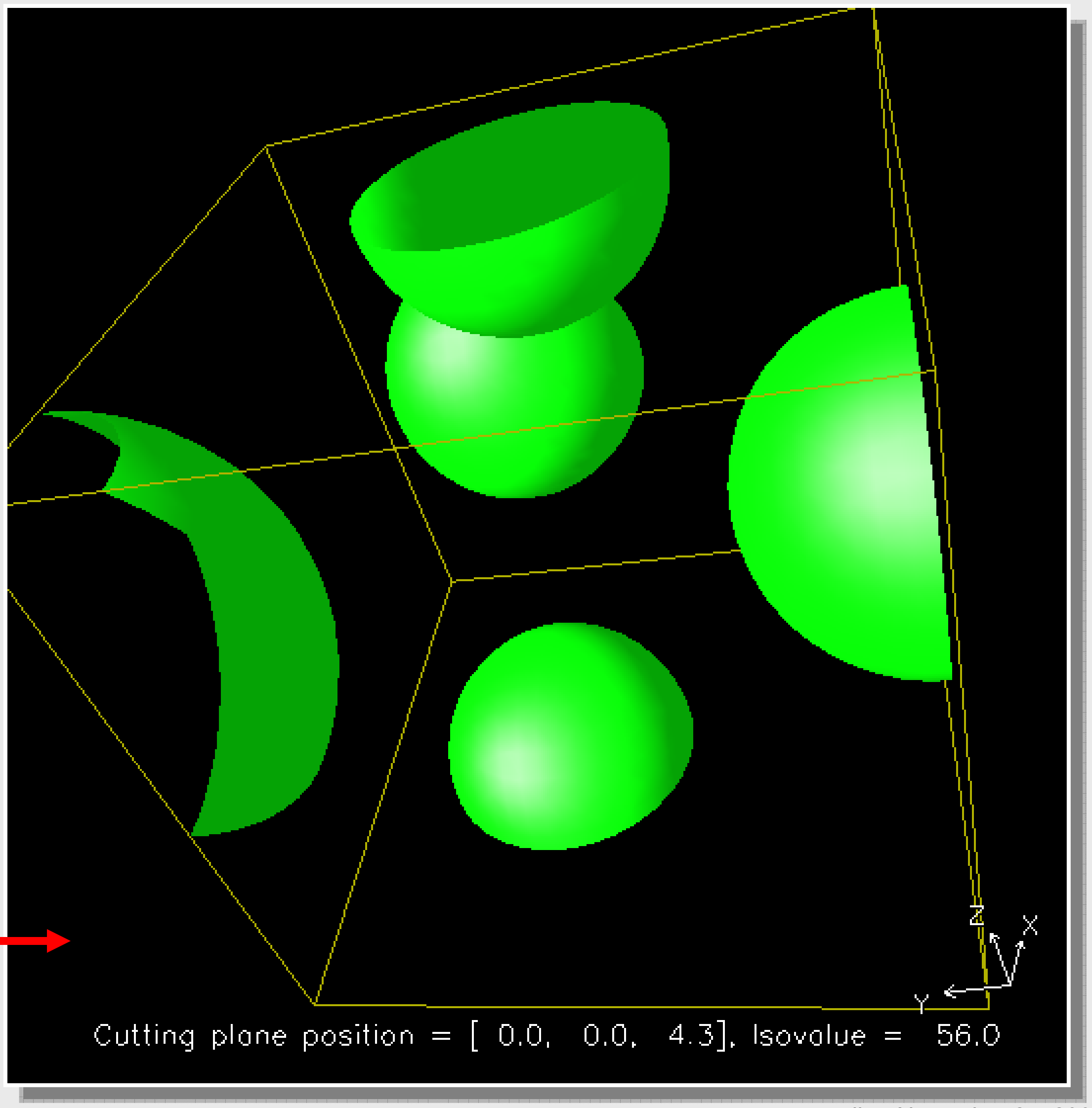
Name	Type	Destination	Cache
output	value, value list, field	Isosurface	All Results

In this case, *Compute* turns an integer into a scalar to be used to animate the isovalue

Oregon State University

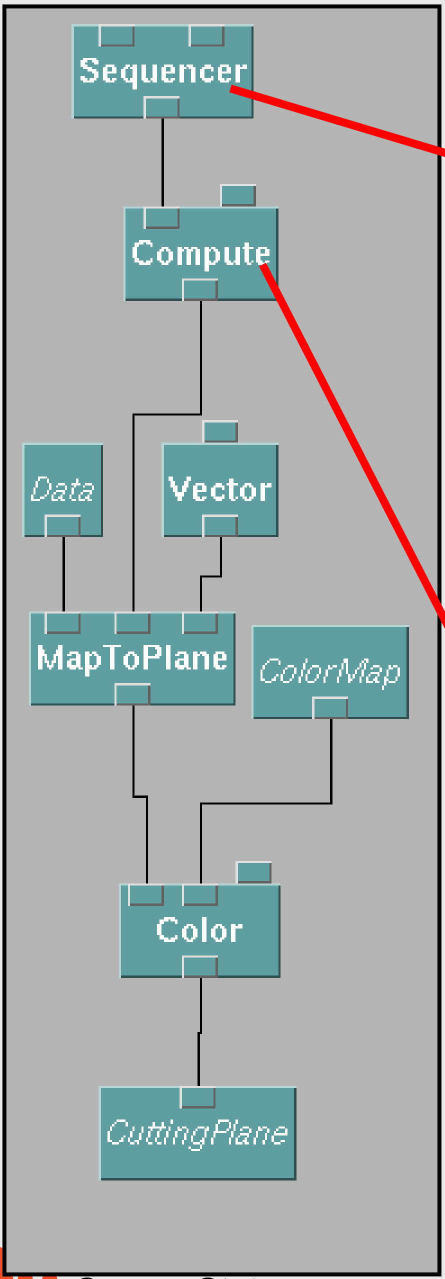
The Sequencer Module: Setting a Scalar Isovalue

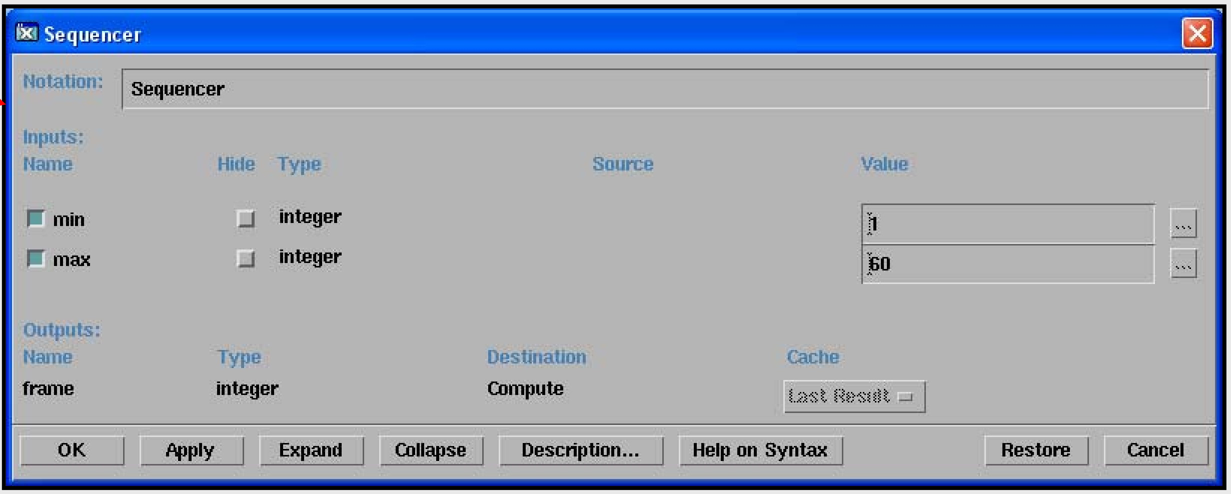




Oregon State University

The Sequencer Module: Setting a Vector to act as a Plane Location





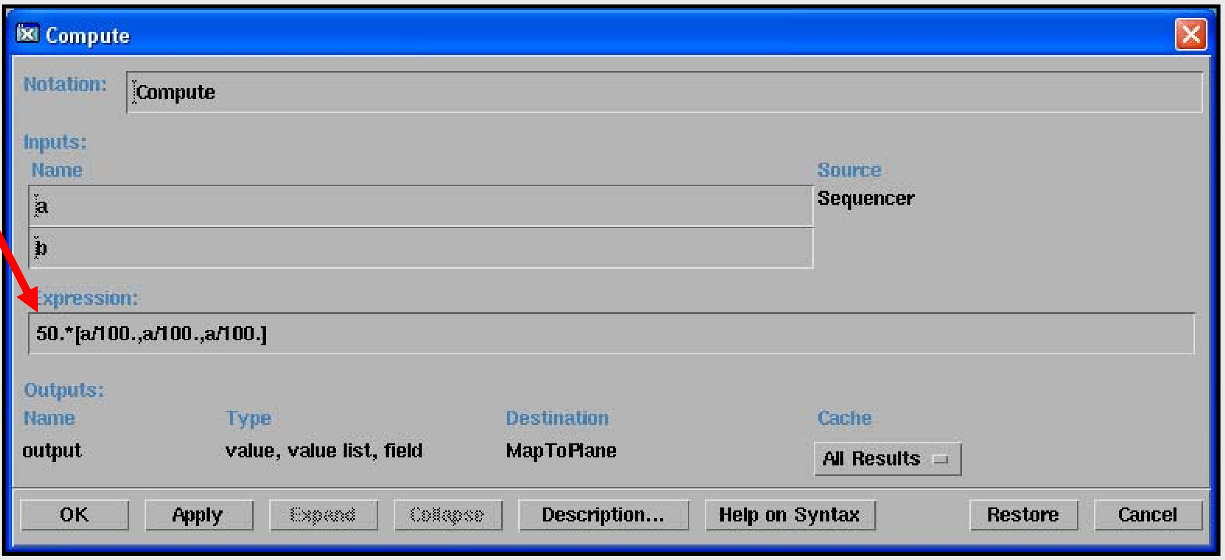
Notation: Sequencer

Inputs:

Name	Hide	Type	Source	Value
min		integer		1
max		integer		60

Outputs:

Name	Type	Destination	Cache
frame	integer	Compute	Last Result



Notation: Compute

Inputs:

Name	Source
a	Sequencer
b	

Expression:

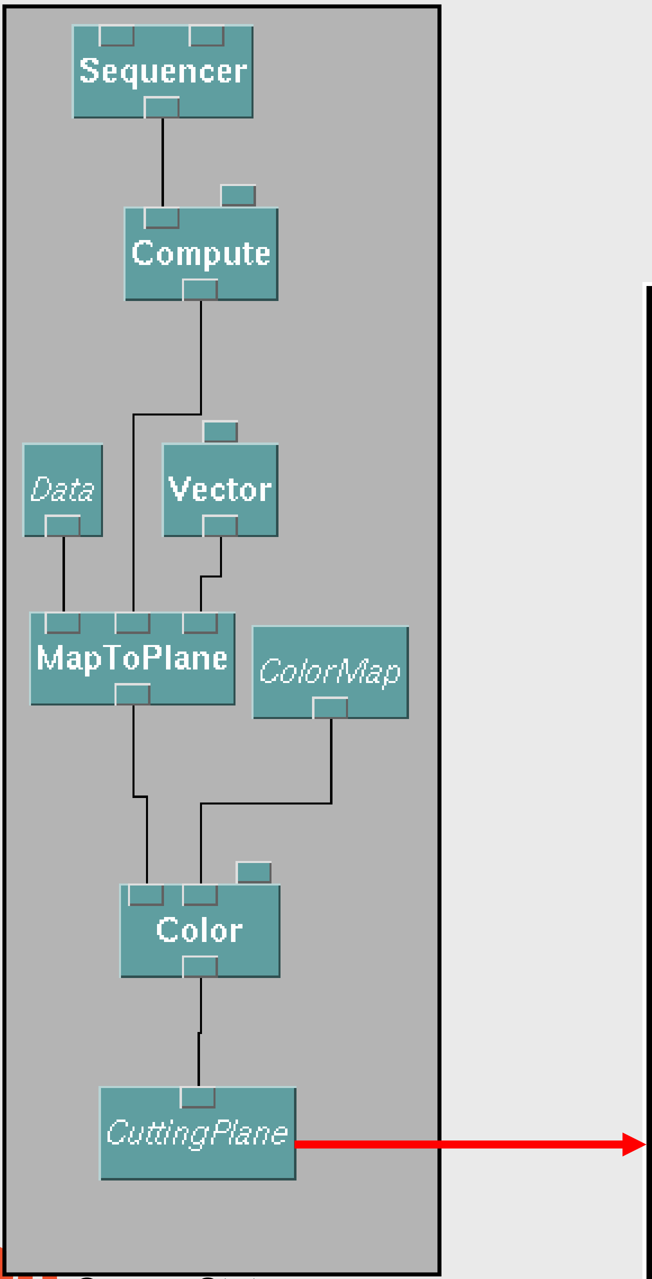
```
50.*[a/100.,a/100.,a/100.]
```

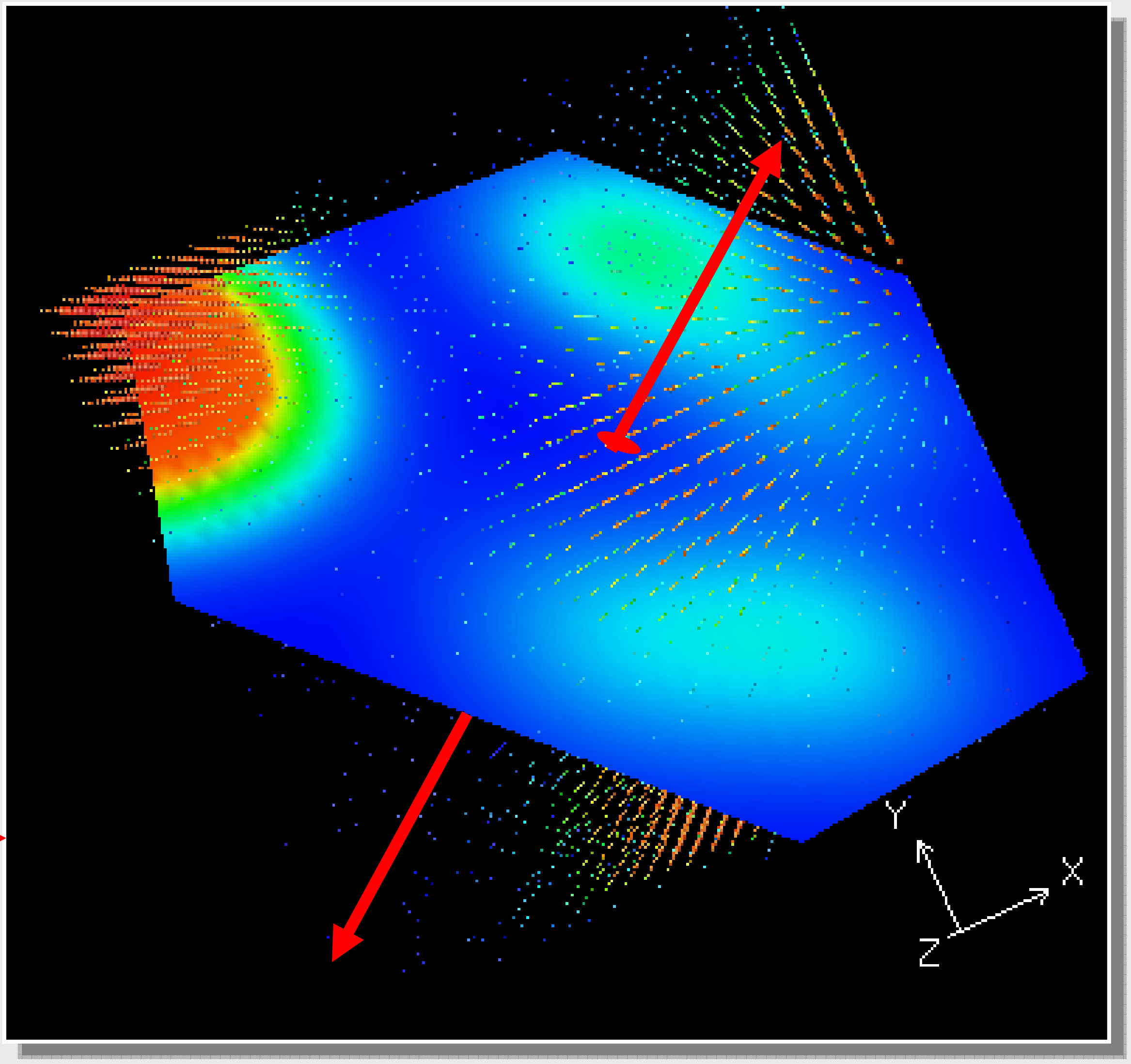
Outputs:

Name	Type	Destination	Cache
output	value, value list, field	MapToPlane	All Results

In this case, *Compute* turns an integer into a 3-element vector to be used to animate the position of the cutting plane

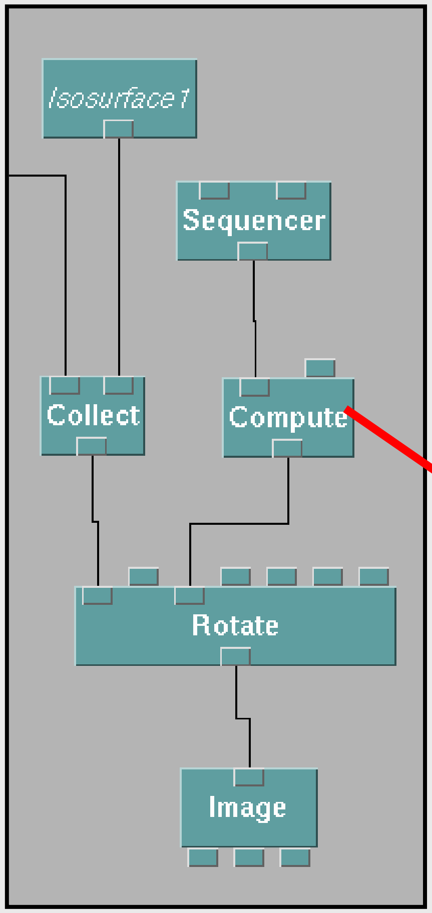
The Sequencer Module: Setting a Vector to act as a Plane Location

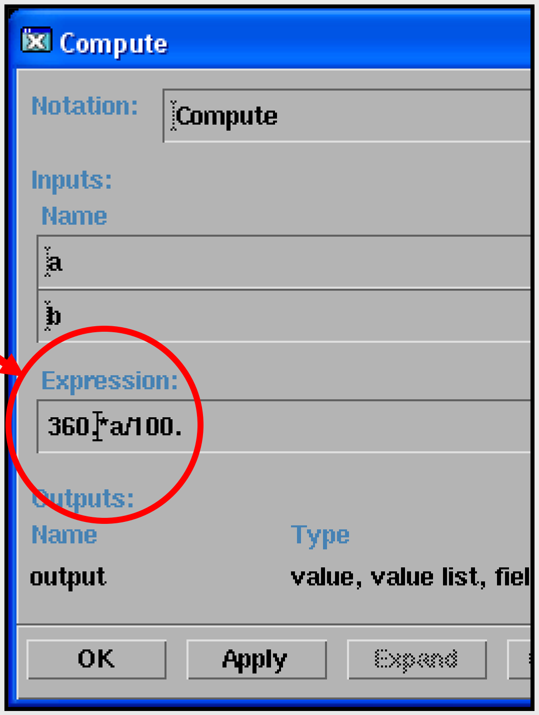




Oregon State University

The Sequencer Module: Setting a Transformation




Compute

Notation: Compute

Inputs:

Name

a

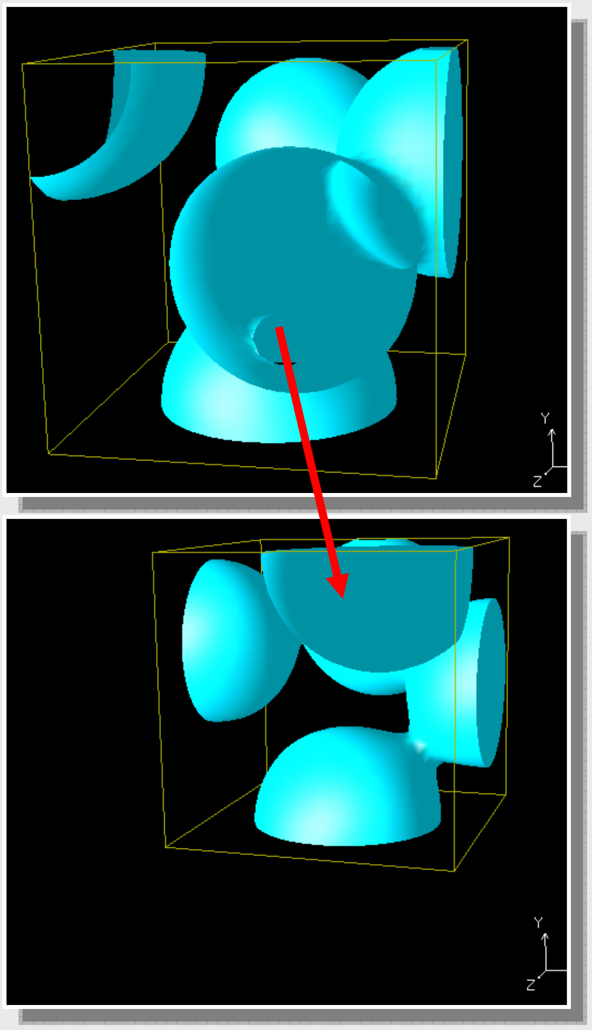
b

Expression:

360.*a/100.

Outputs:

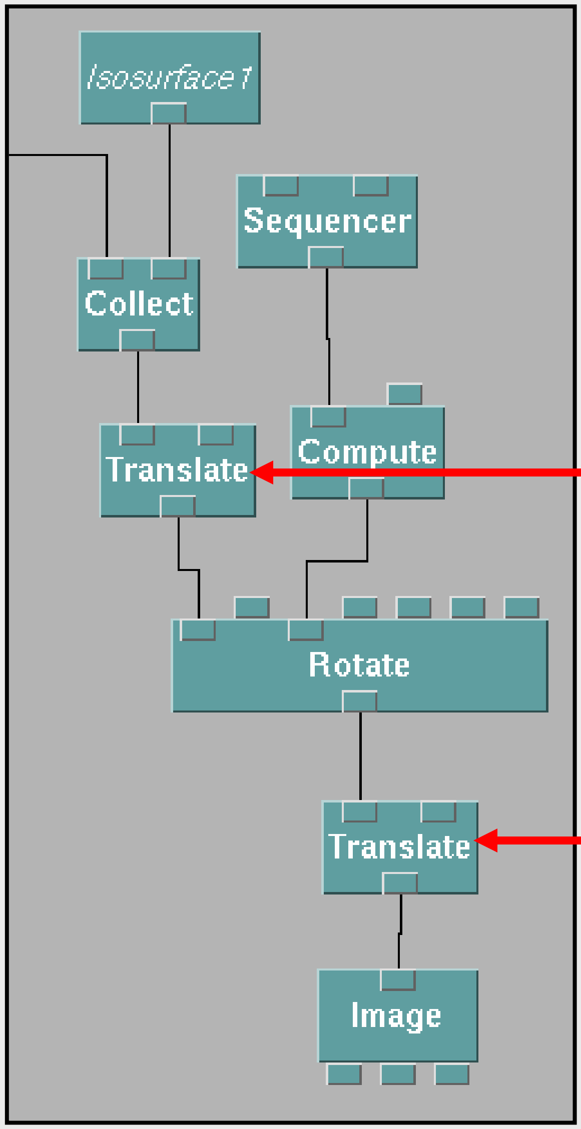
Name	Type
output	value, value list, fiel



In this case, *Compute* turns an integer into a rotation angle in *degrees*.

Oregon State University

Why Does the Rotation Occur around the Edge of the Cube, not about its Center?

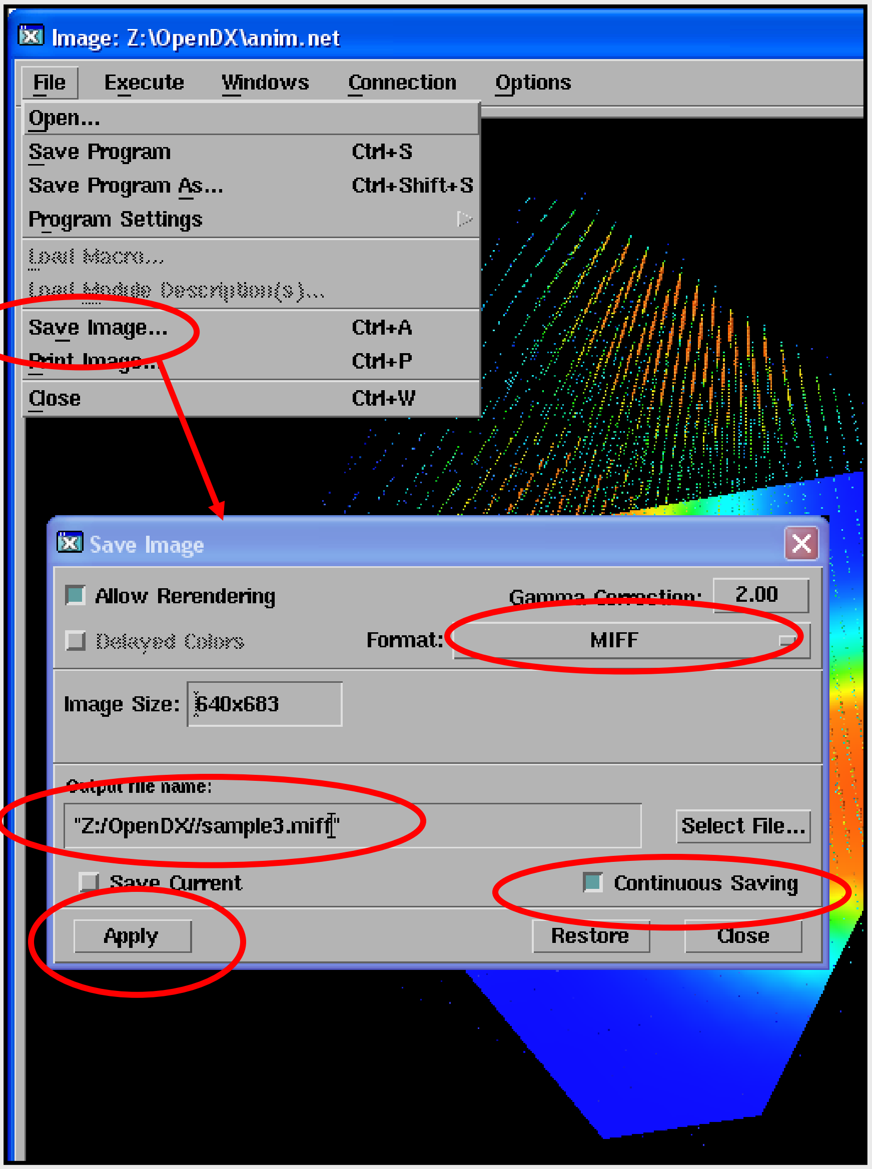


Rotation and Scaling *always occur about the origin*. To change this to the center of the volume, translate the volume to the origin, perform the rotation or scale, and then translate it back.

Translate by [-15,-15,-15]

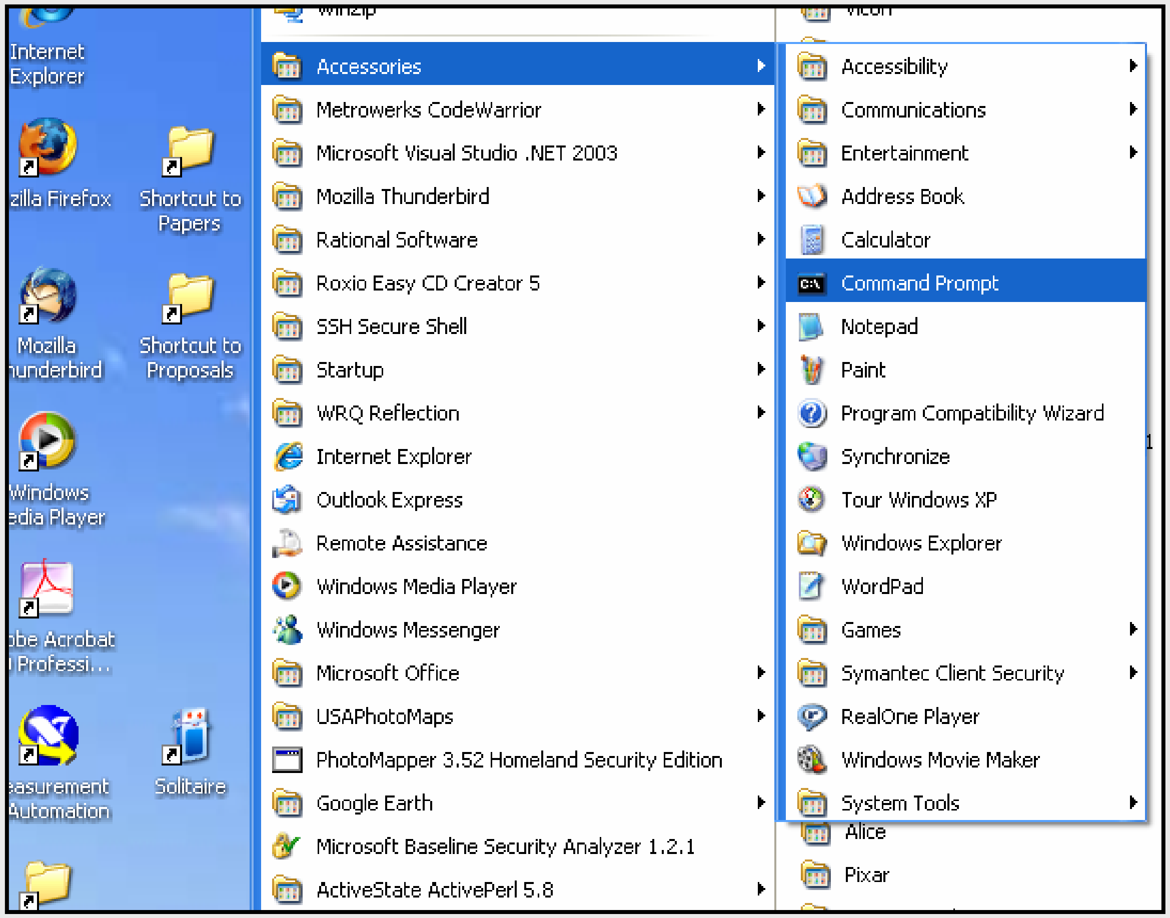
Translate by [15,15,15]

Writing Out a MIFF Animation File



[✓ Allow Rerendering]
Gamma Correction: 2.00
Delayed Colors
Format: MIFF
Image Size: 640x683
Output file name: "Z:/OpenDX//sample3.miff"
Save Current
[✓ Continuous Saving]
Apply

Converting a MIFF Animation File into an Animated GIF File using the ImageMagick Package

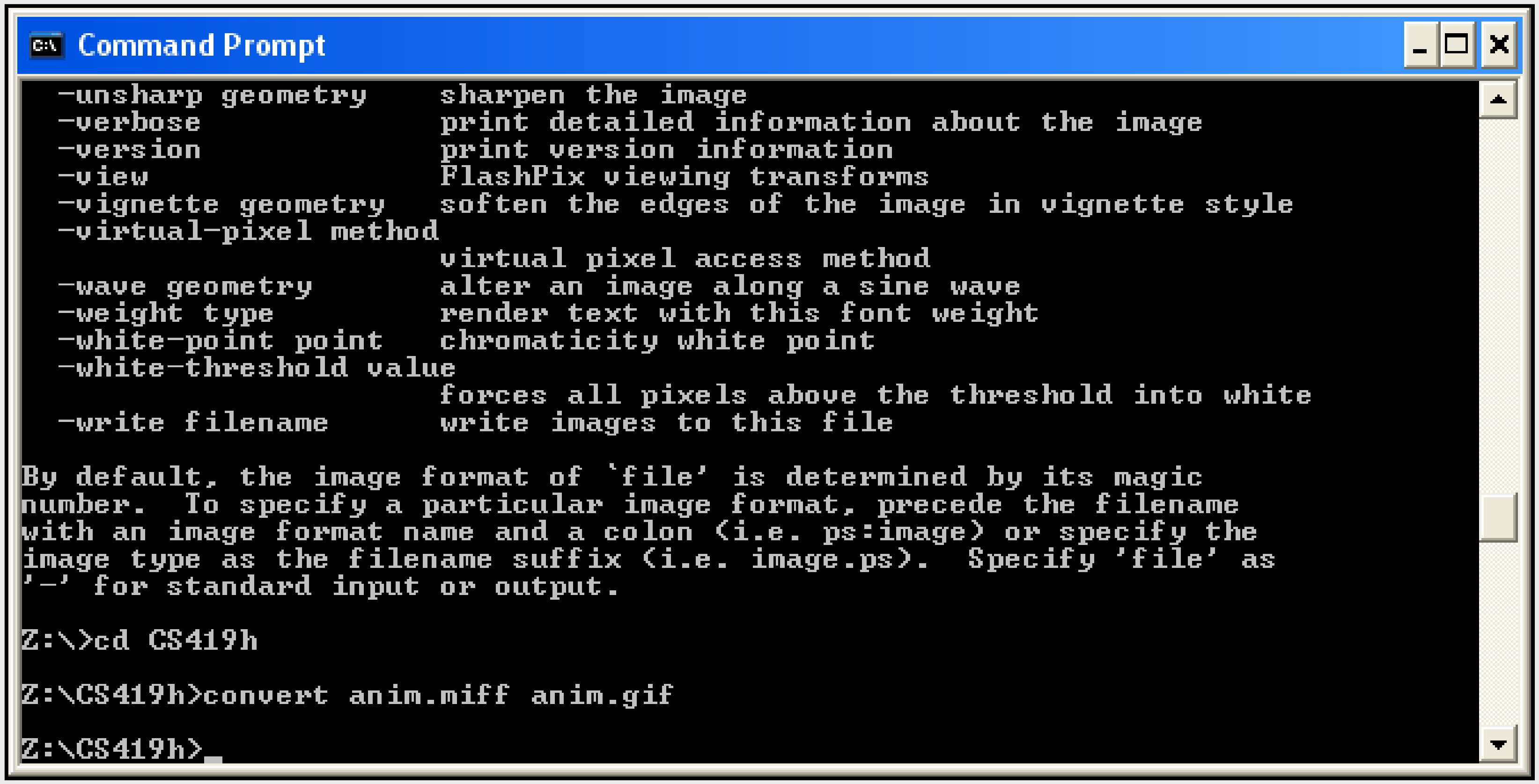


Click on: **Start→All Programs →Accessories →Command Prompt**

Oregon State University

Converting a MIFF Animation File into an Animated GIF File using the ImageMagick Package

Type: **convert anim.miff anim.gif**
(where anim is the name of your MIFF animation file written from the *Image* module)



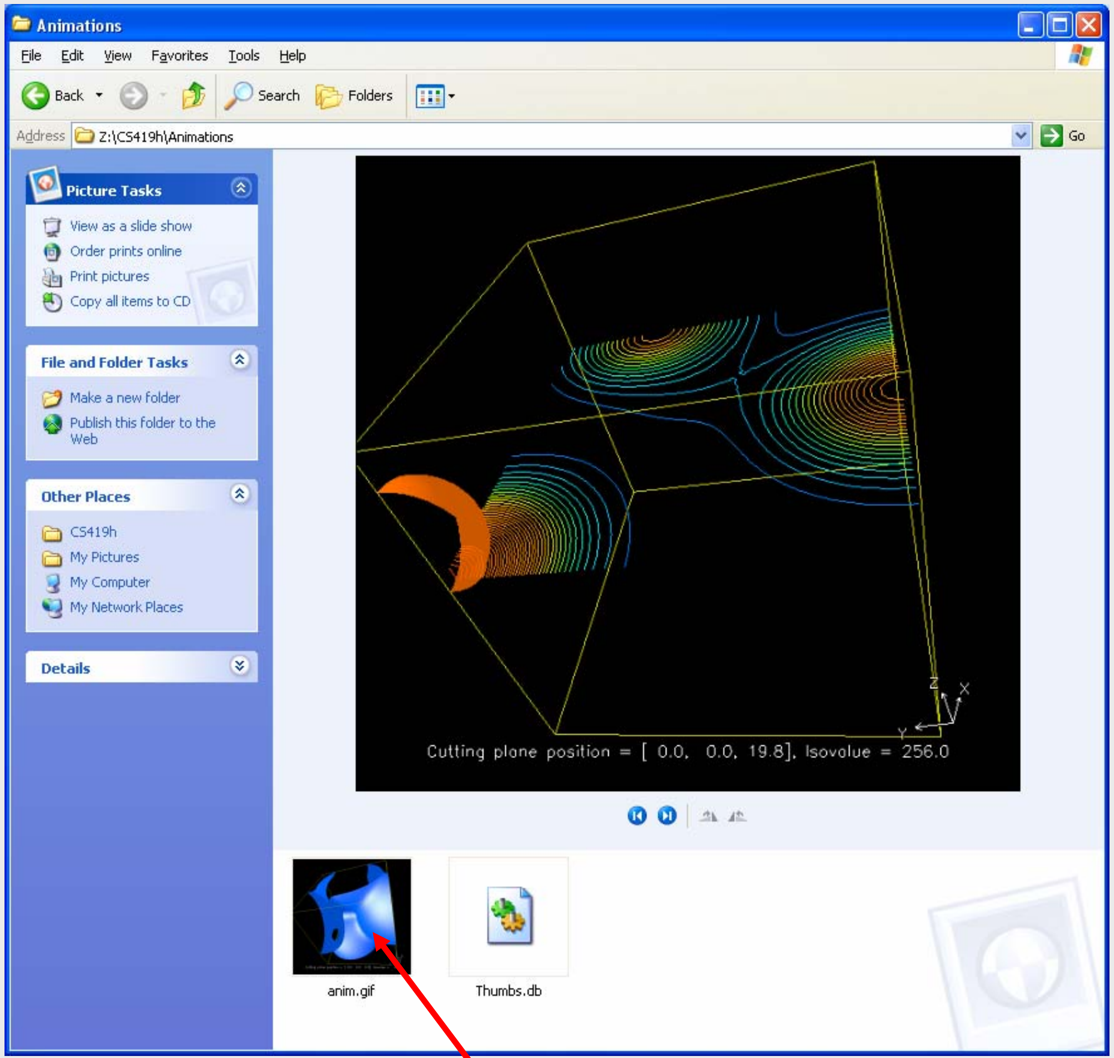
Command Prompt

```
  -unsharp geometry    sharpen the image
  -verbose             print detailed information about the image
  -version             print version information
  -view                FlashPix viewing transforms
  -vignette geometry   soften the edges of the image in vignette style
  -virtual-pixel method
                       virtual pixel access method
  -wave geometry       alter an image along a sine wave
  -weight type         render text with this font weight
  -white-point point   chromaticity white point
  -white-threshold value
                       forces all pixels above the threshold into white
  -write filename      write images to this file

By default, the image format of `file' is determined by its magic
number.  To specify a particular image format, precede the filename
with an image format name and a colon (i.e. ps:image) or specify the
image type as the filename suffix (i.e. image.ps).  Specify 'file' as
'-' for standard input or output.

Z:\>cd CS419h

Z:\CS419h>convert anim.miff anim.gif

Z:\CS419h>_
```

http://www.imagemagick.org

OSU Oregon State University

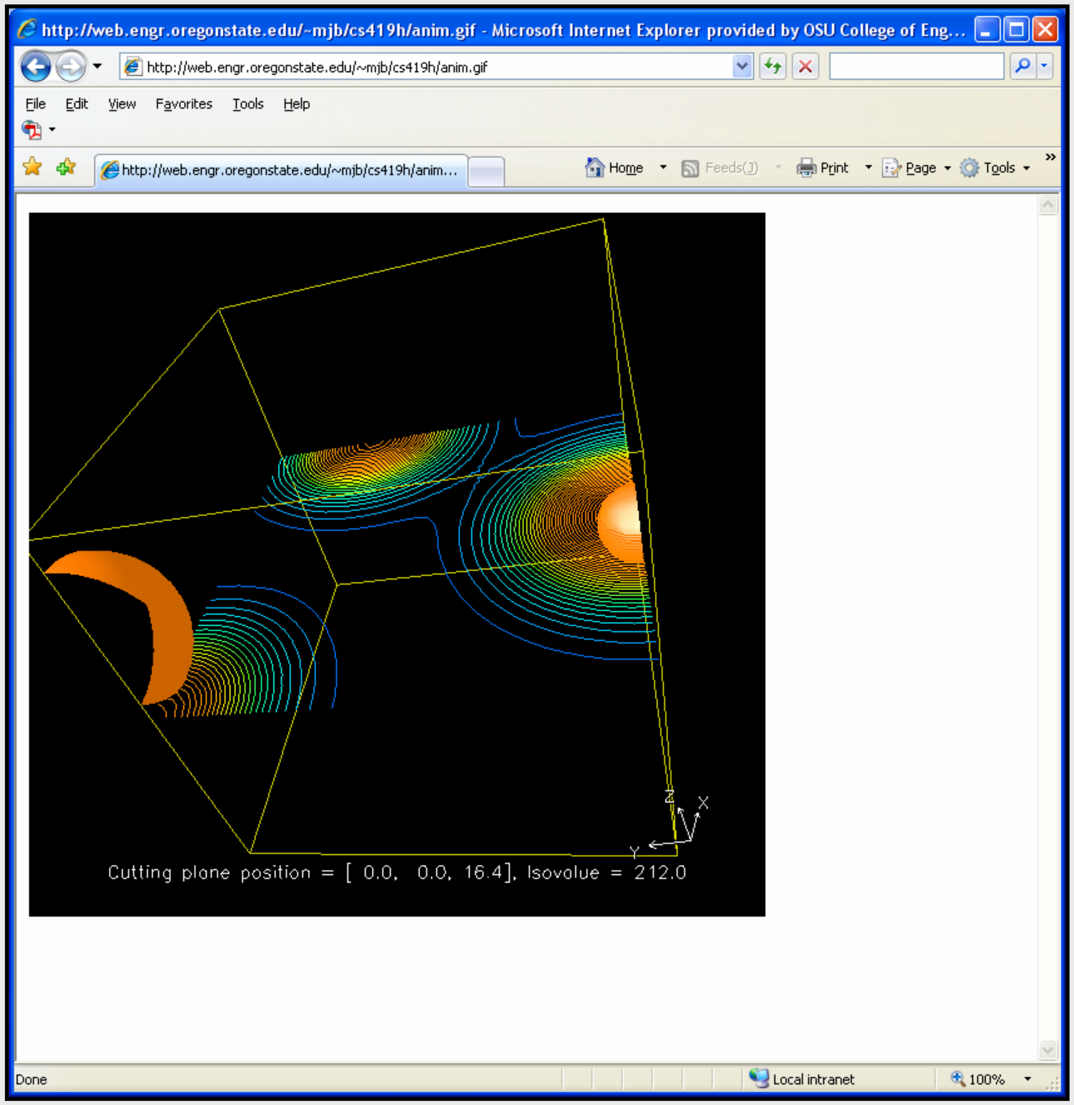
Animated GIF Files work in Windows Explorer



Double-click on the animated GIF file

Oregon State University

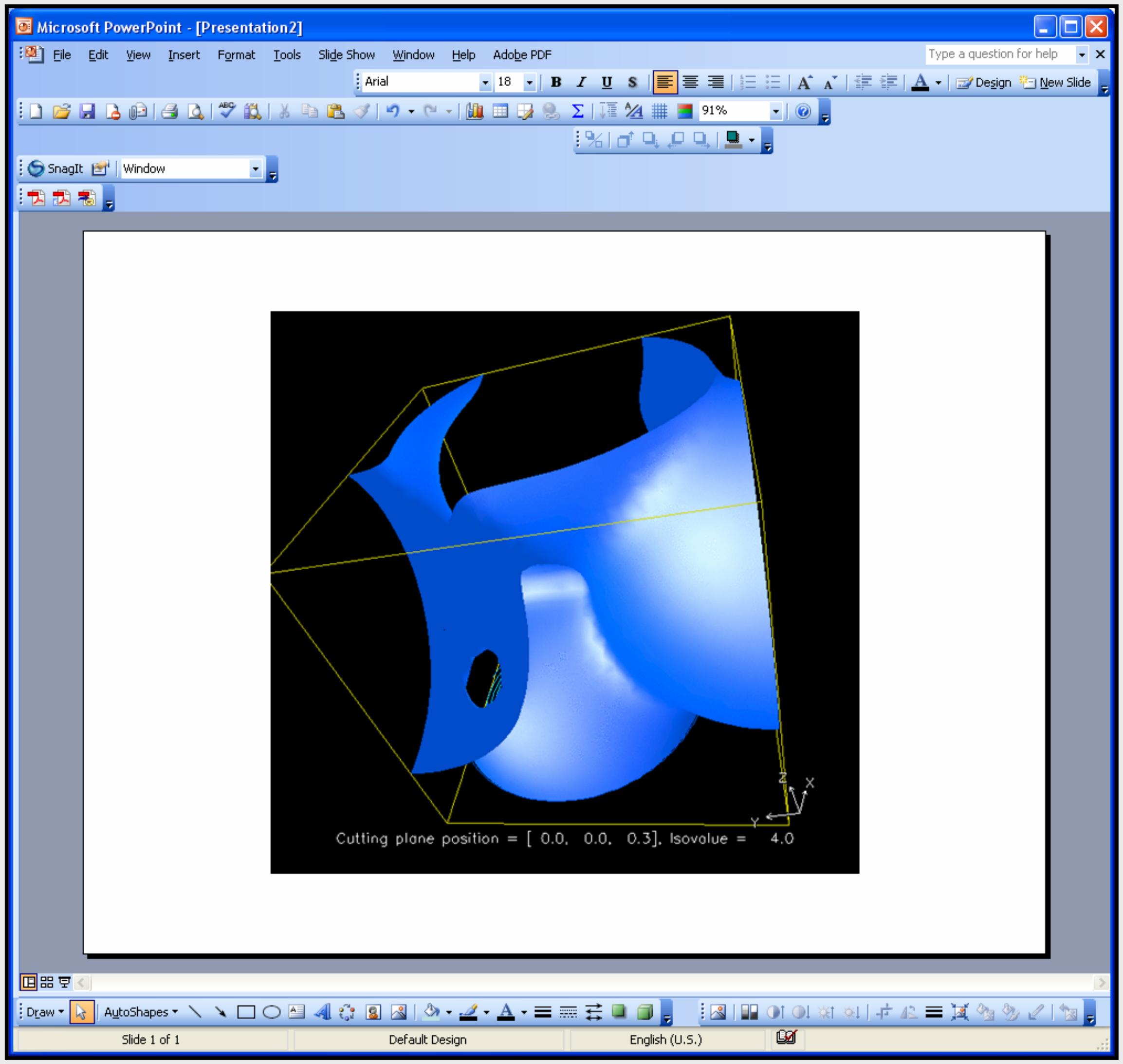
Animated GIF Files work in Web Pages


http://web.engr.oregonstate.edu/~mjb/cs419h/anim.gif - Microsoft Internet Explorer provided by OSU College of Eng...
http://web.engr.oregonstate.edu/~mjb/cs419h/anim.gif
File Edit View Favorites Tools Help
Home Feeds(J) Print Page Tools
Z X Y
Cutting plane position = [0.0, 0.0, 16.4], Isovalue = 212.0
Done
Local intranet
100%

```
<img src="anim.gif">
```

Oregon State University

Animated GIF Files work in PowerPoint


Microsoft PowerPoint - [Presentation2]
File Edit View Insert Format Tools Slide Show Window Help Adobe PDF
Type a question for help
Arial
18
Design
New Slide
91%
SnagIt
Window
Draw
AutoShapes
Slide 1 of 1
Default Design
English (U.S.)
Cutting plane position = [0.0, 0.0, 0.3], Isovalue = 4.0
Z
X
Y

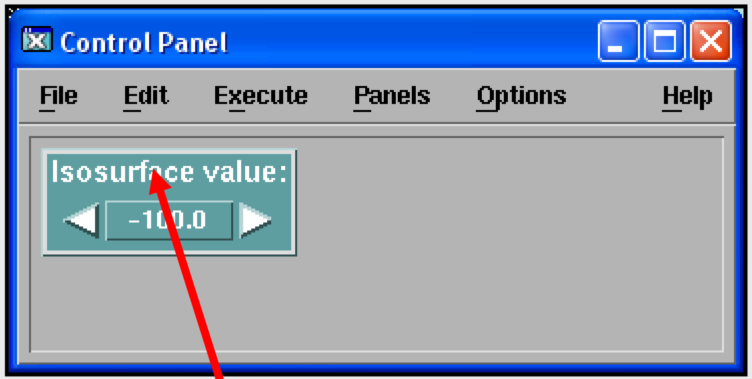
Insert→Picture →From File…

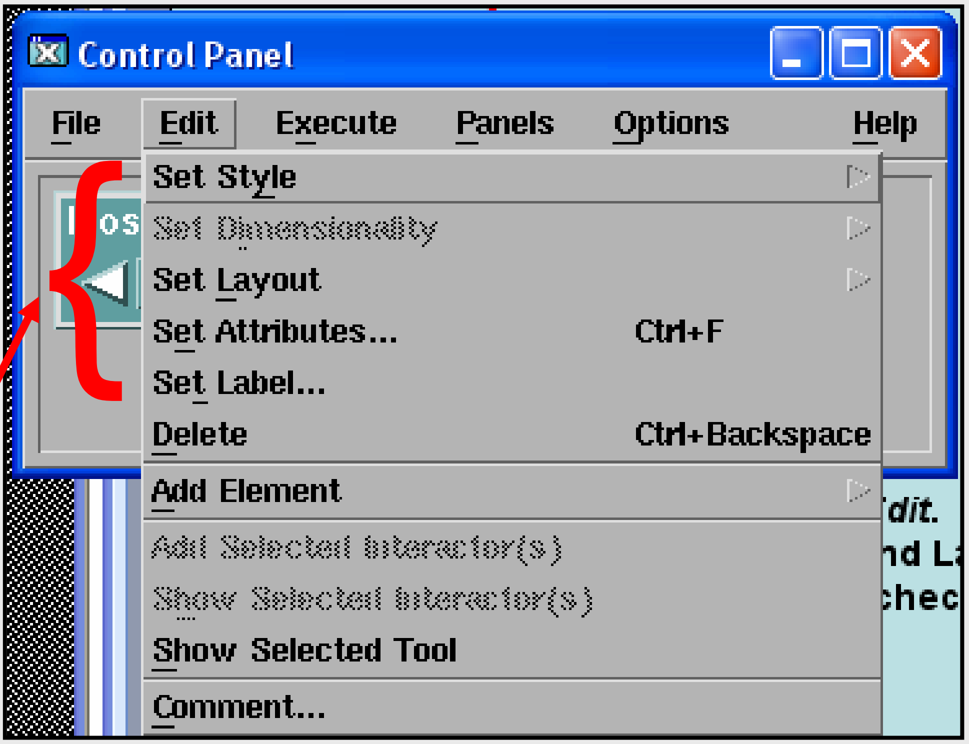
In a presentation, the image will start animating when this slide becomes active

Oregon State University

Interactors

Editing Interactor Attributes



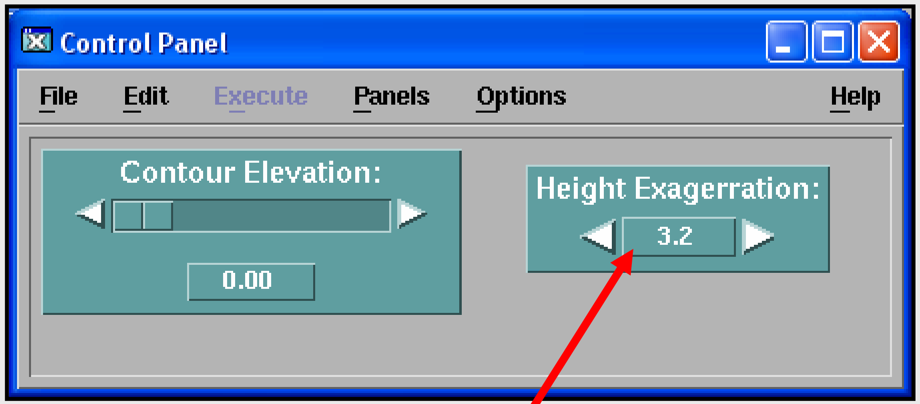


Select an Interactor by *Left-Clicking* its Label.

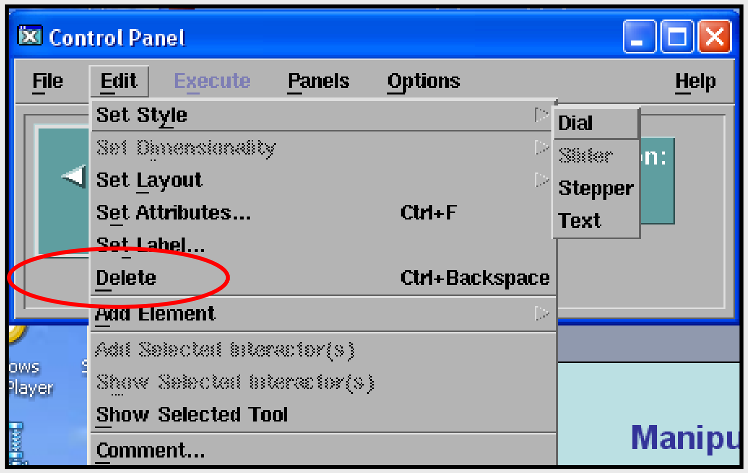
Then, click *Edit*. You can change the Interactor's Style, Layout, Attributes, and Label. Under *Set Attributes*, clicking on the *Continuous* checkbox is usually a good thing.

Oregon State University

Ganging Interactors

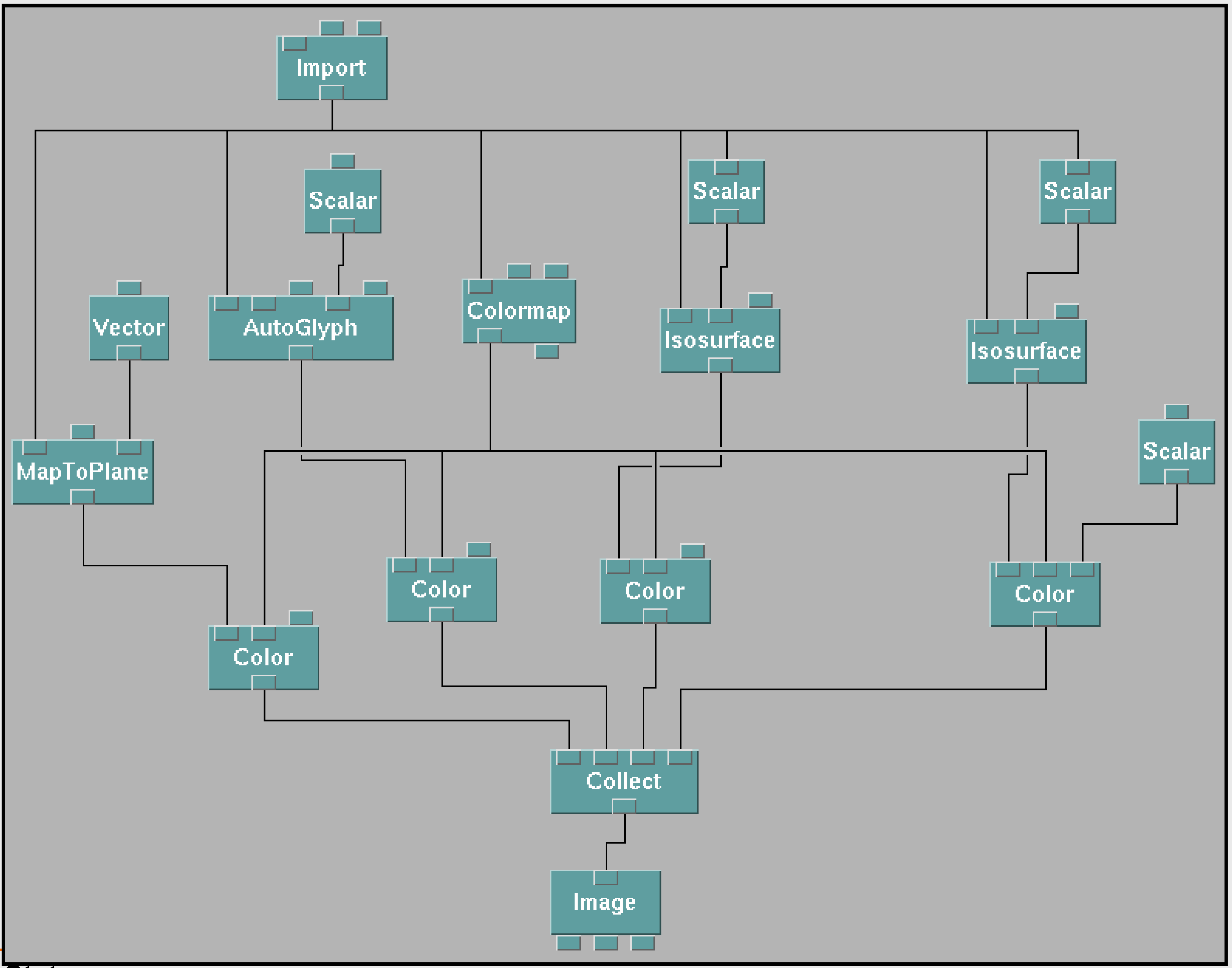


You can place all Interactors in a single window by using the **middle mouse button** to drag them over. This copies them, not moves them. Then select the original Interactor in its original window and *Edit-Delete* it.

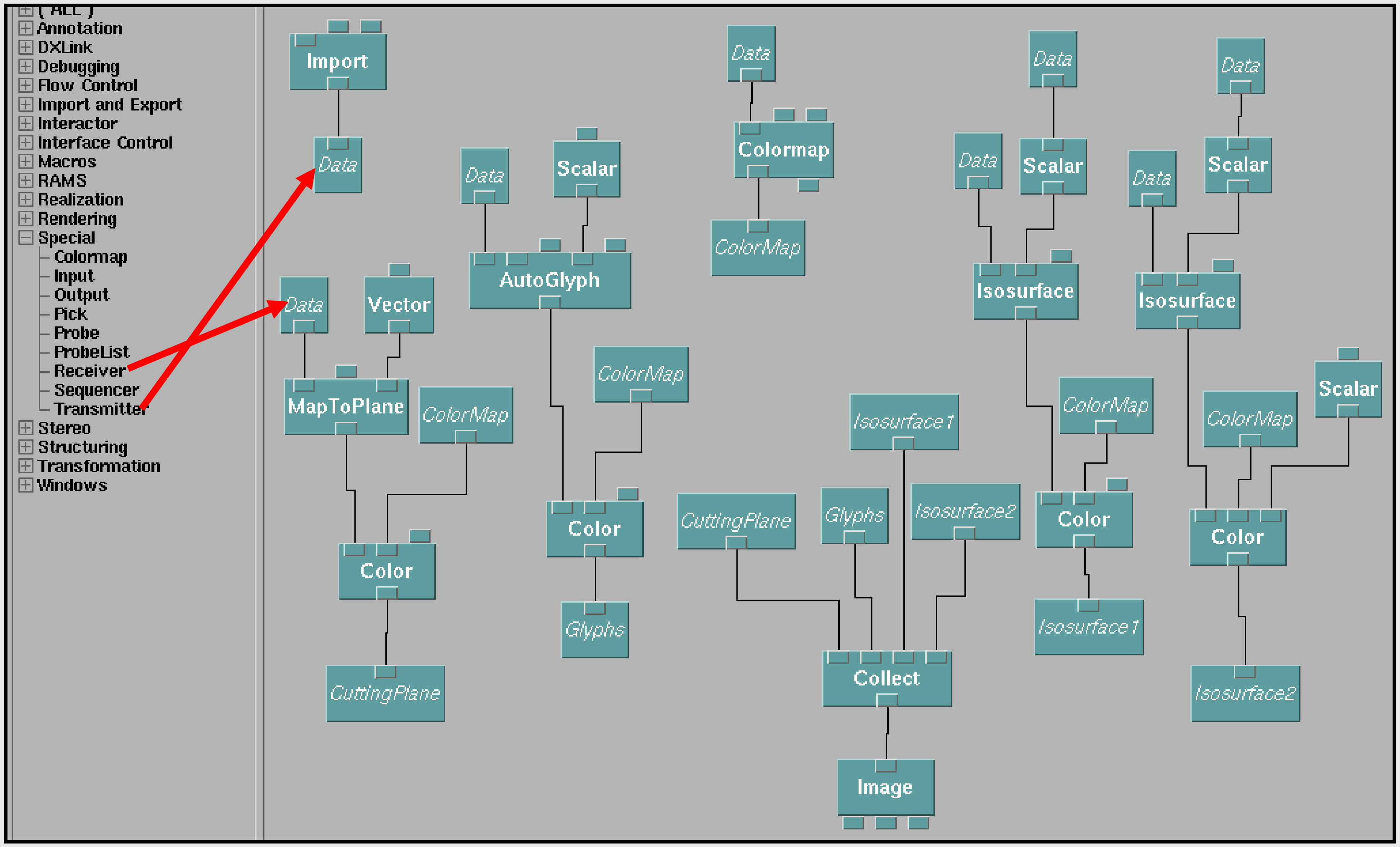


Transmitters and Receivers

It’s Easy to Get Cluttered, Especially Around *Import* and *ColorMap*!



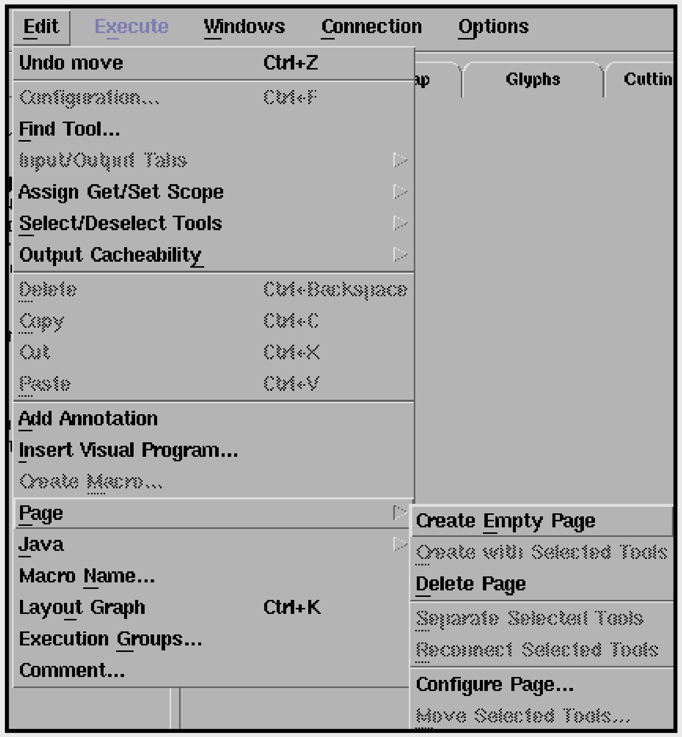
It's Also Easy to Get Un-cluttered with *Transmitter* and *Receiver*



Notice how this lets you create separate "regions" for different functions. Wouldn't it be nice if you could put each region on its own page?

Oregon State University

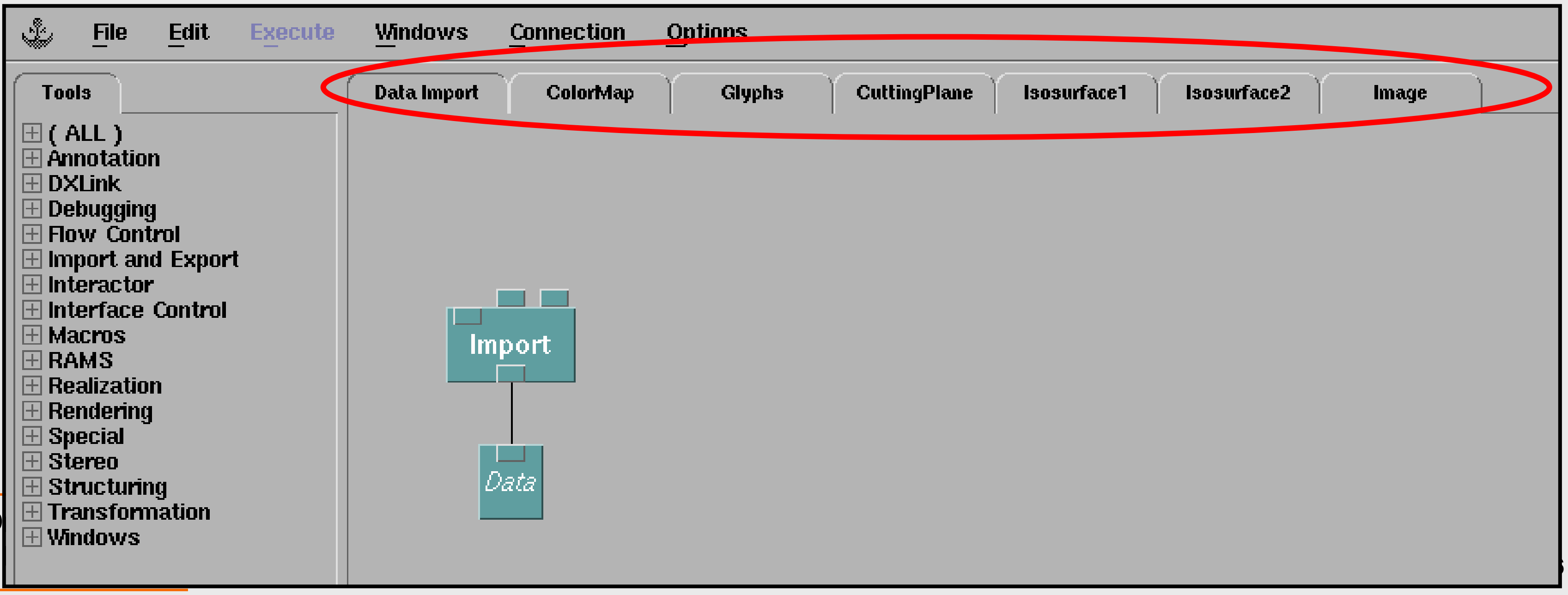
Using *Transmitter* and *Receiver*, You Can Also Spread the Network Out on Multiple "Pages"



Click *Edit→Page →Create Empty Page* to make a new page

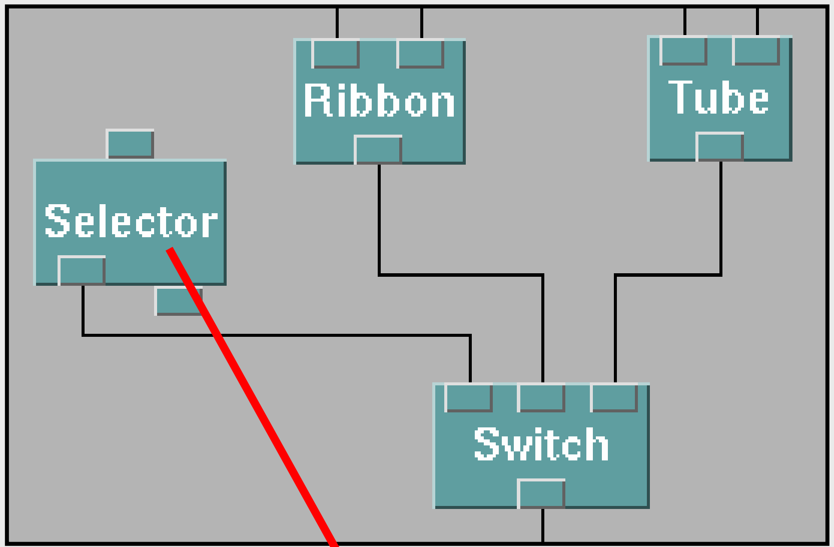
Double-click on the page's tab, type in the page's name, and hit Enter

You can create from scratch in the other pages, or cut-and-paste from where you started

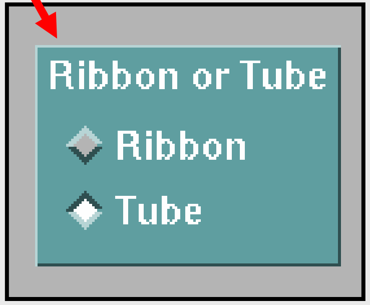


Switching, Selecting, and Toggling

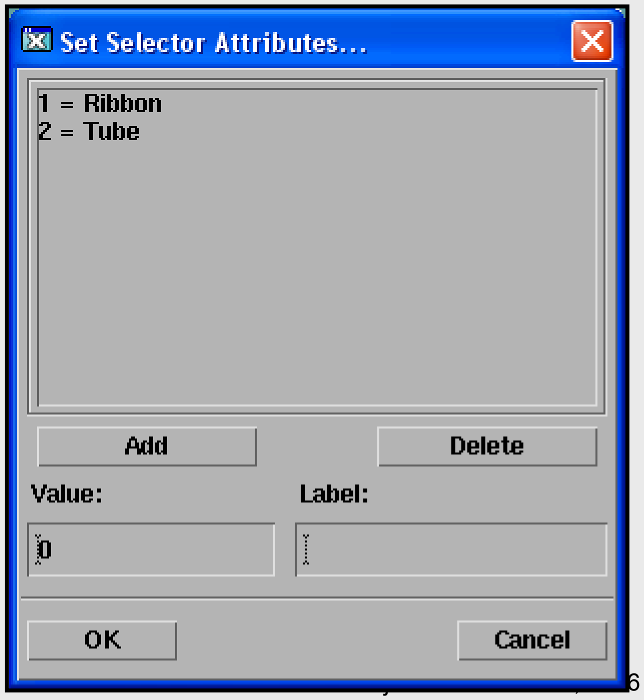
Selecting from Multiple Objects: *Selector* and *Switch*



The Switch module sends nothing through when its first input is 0. It sends the second input through when the first input is 1. It sends the third input through when the first input is 2, etc.



Edit→Set Attributes

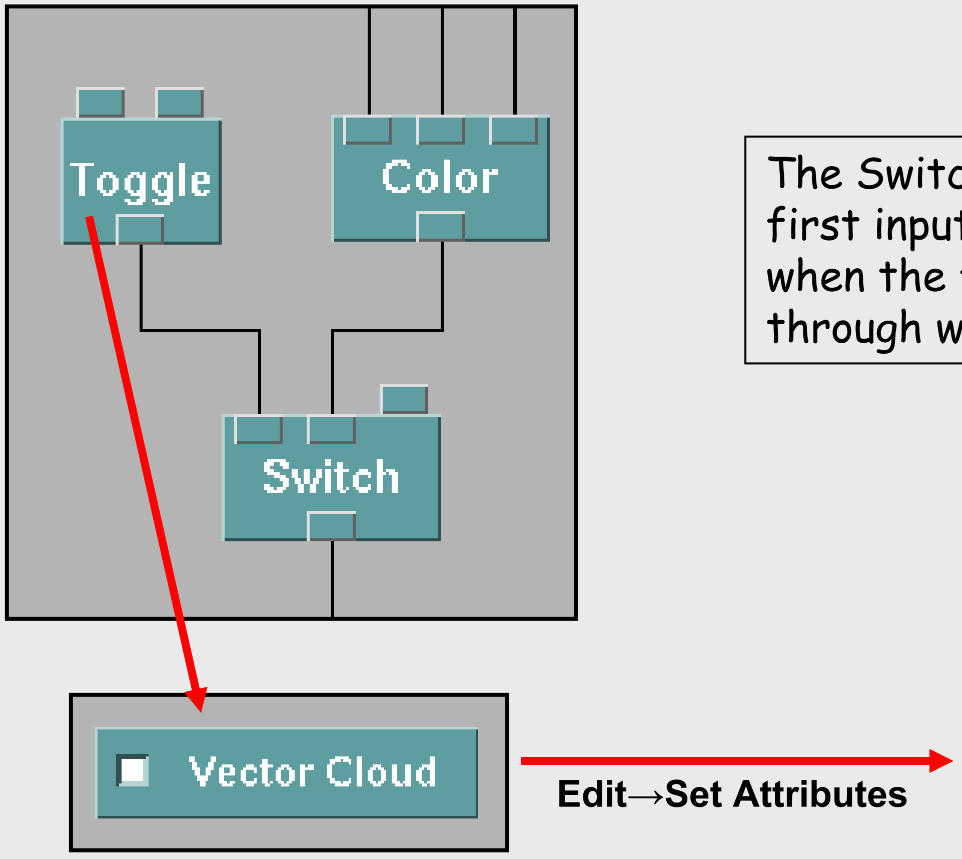


1 = Ribbon
2 = Tube

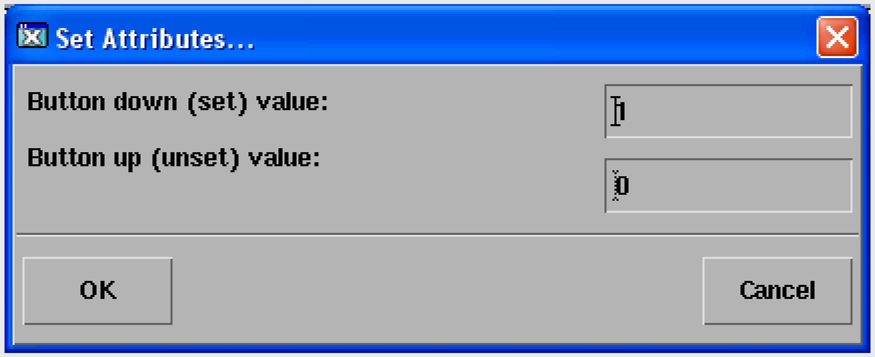
Value: 0

Label:

Toggling Objects On and Off : *Toggle* and *Switch*



The Switch module sends nothing through when its first input is 0. It sends the second input through when the first input is 1. It sends the third input through when the first input is 2, etc.

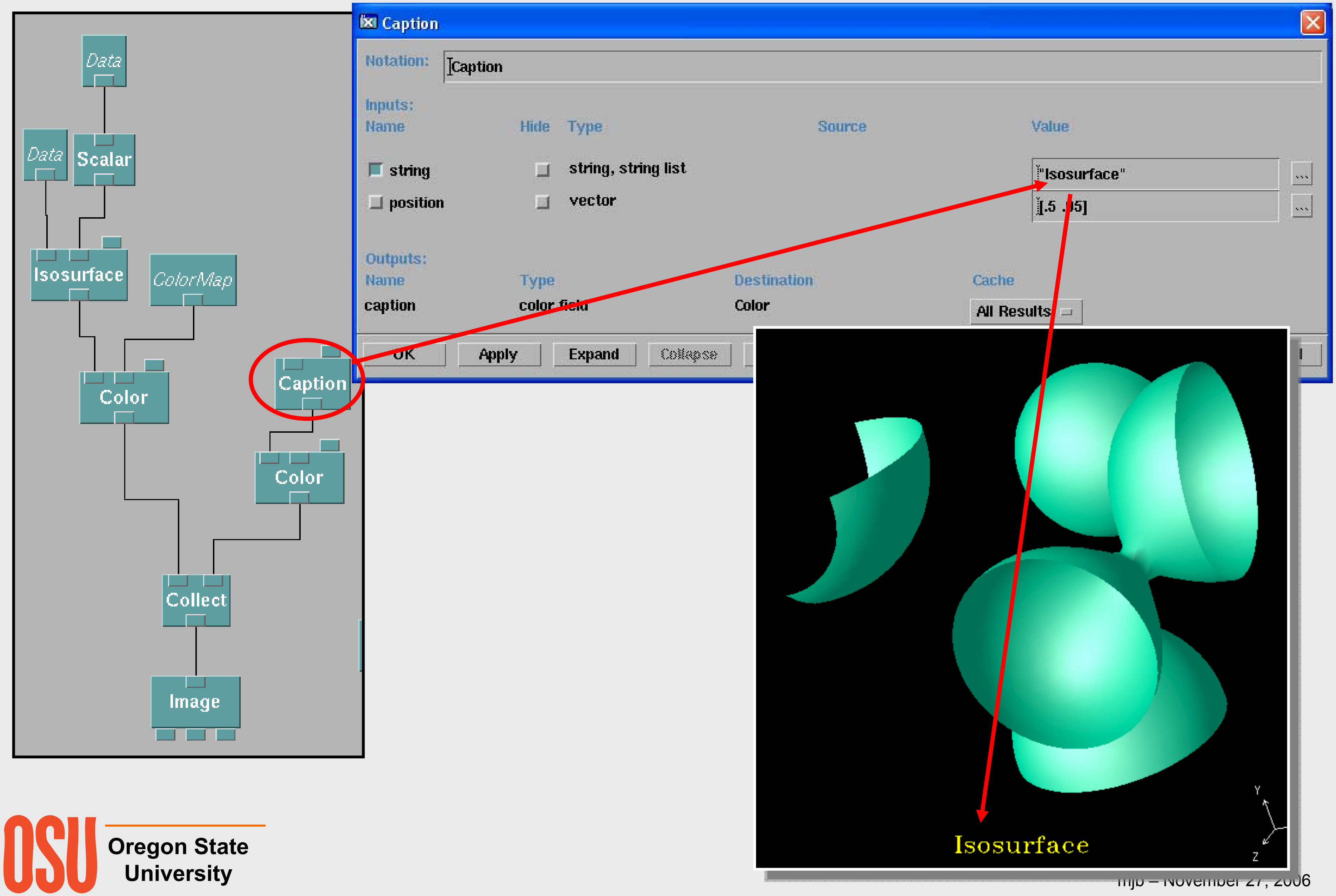


Button down (set) value: 1

Button up (unset) value: 0

Captions

Placing a Caption on a Visualization



Caption

Notation: Caption

Inputs:

Name	Hide	Type	Source	Value
string		string, string list		"Isosurface"
position		vector		[.5 .05]

Outputs:

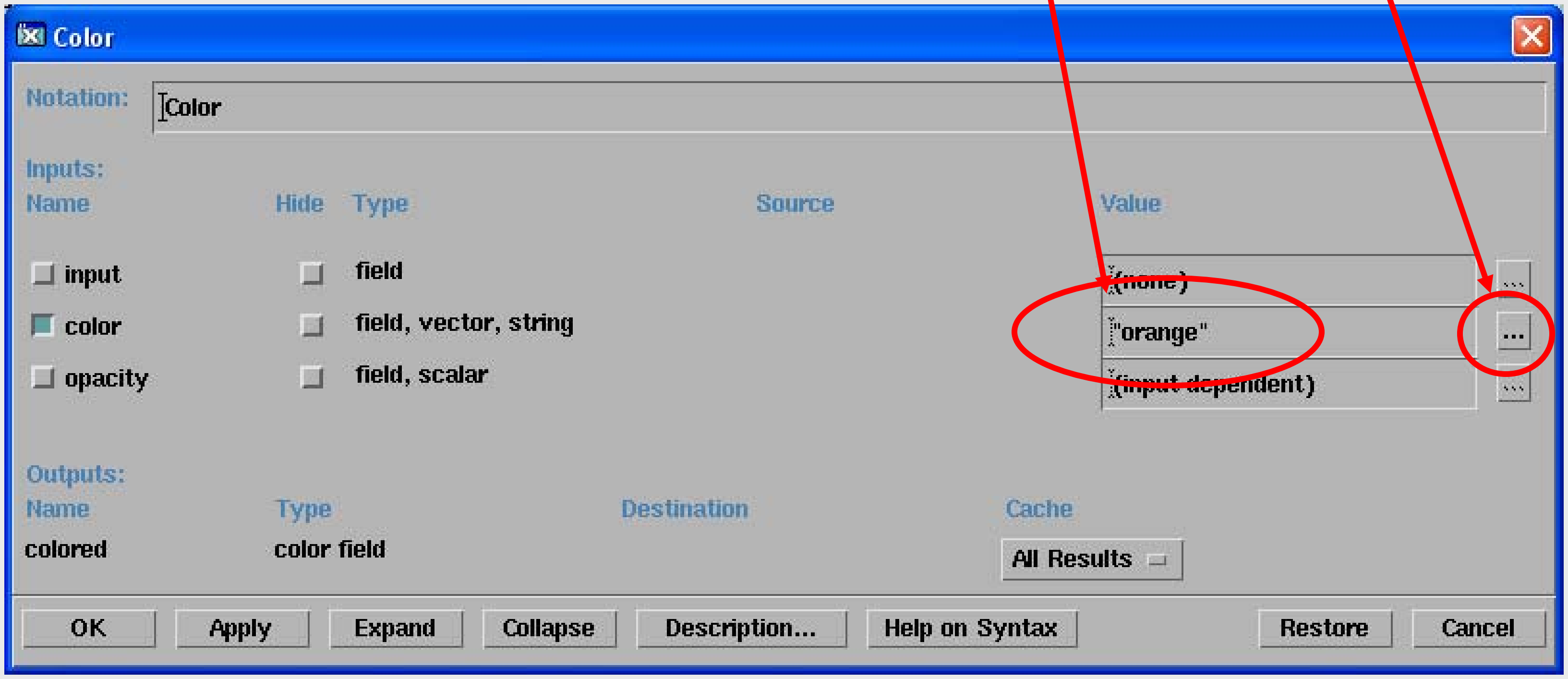
Name	Type	Destination	Cache
caption	color field	Color	All Results

OK Apply Expand Collapse

Oregon State University

Setting the Color of the Caption

Click here to get a list of simple colors, or type the name of a color here



Color

Notation: Color

Inputs:

Name	Hide	Type	Source	Value
☐ input	☐	field		(none)
■ color	☐	field, vector, string		"orange"
☐ opacity	☐	field, scalar		(input dependent)

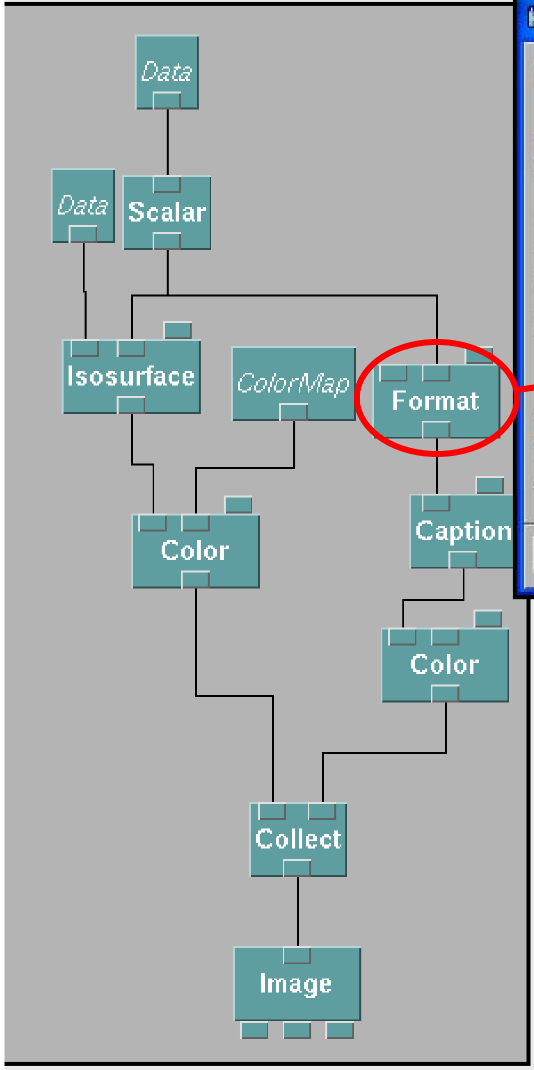
Outputs:

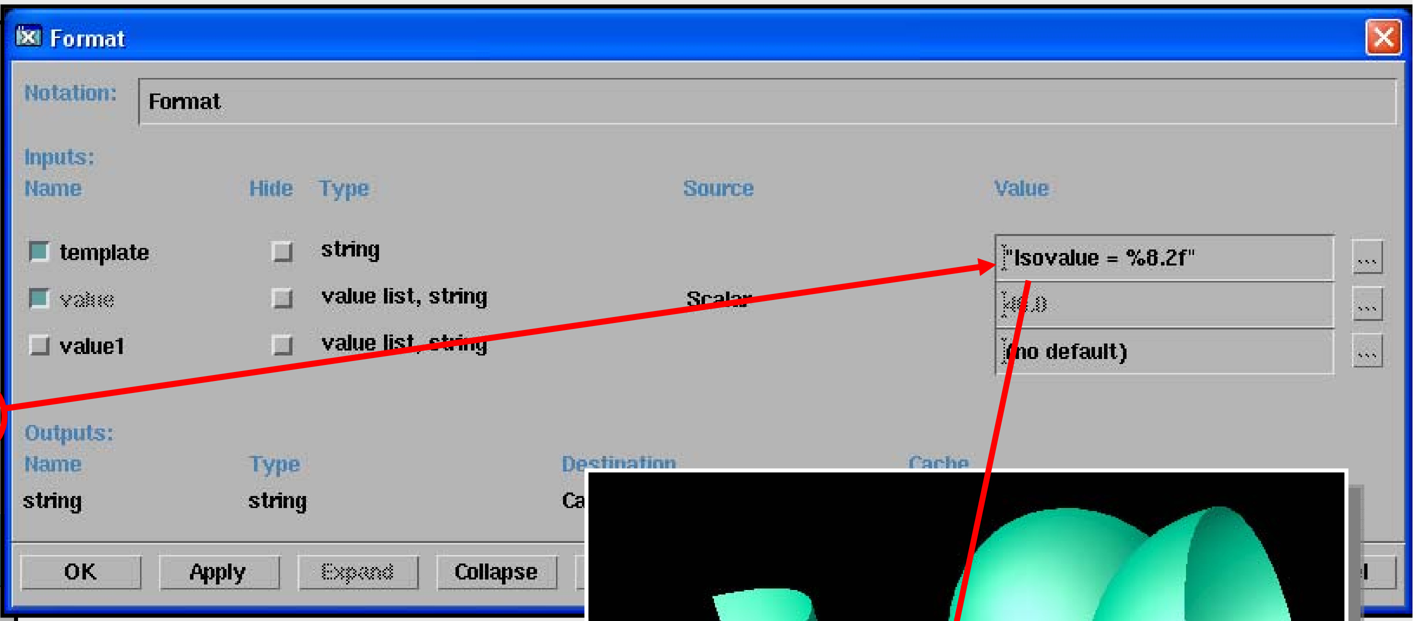
Name	Type	Destination	Cache
colored	color field		All Results

Color Names you can use in the Color Module

aquamarine	darkturquoise	lightgrey	midnightblue	springgreen
black	dimgray	lightsteelblue	navy	steelblue
blue	dimgrey	limegreen	navyblue	tan
blueviolet	firebrick	magenta	orange	thistle
brown	forestgreen	maroon	orangered	turquoise
cadetblue	gold	mediumaquamarine	orchid	violet
coral	goldenrod	mediumblue	palegreen	violetred
cornflowerblue	gray	mediumforestgreen	pink	wheat
cyan	green	mediumgoldenrod	plum	white
darkgreen	greenyellow	mediumorchid	red	yellow
darkoliveggreen	grey	mediumseagreen	salmon	yellowgreen
darkorchid	indianred	mediumslateblue	seagreen	
darkslateblue	khaki	mediumspringgreen	sienna	
darkslategray	lightblue	mediumturquoise	skyblue	
darkslategrey	lightgray	mediumvioletred	slateblue	

Placing a Data-formatted Caption on a Visualization





Format

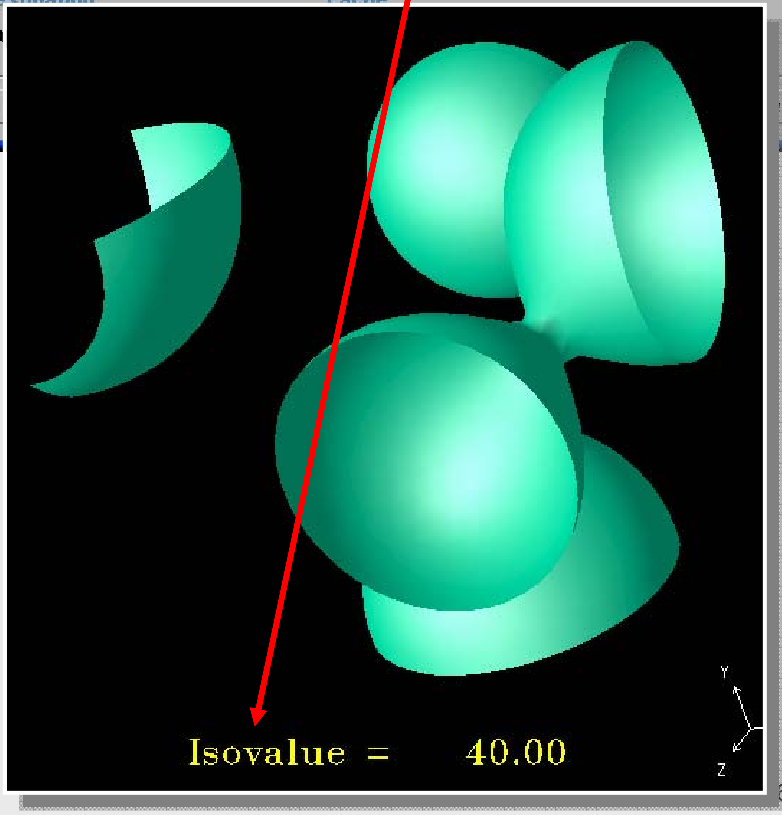
Notation: Format

Inputs:

Name	Hide	Type	Source	Value
template		string		"Isovalue = %8.2f"
value		value list, string	Scalar	40.0
value1		value list, string		(no default)

Outputs:

Name	Type	Destination	Cache
string	string	Ca	



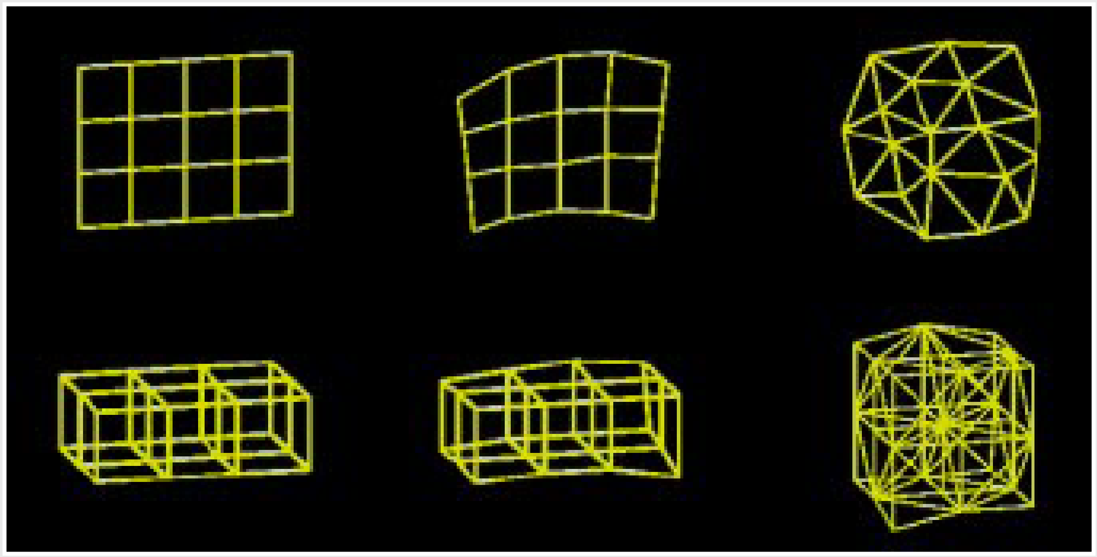
Oregon State University

Importing Your Own Data

OpenDX Data Grid Types

Surfaces

Volumes



Regular:
R positions,
R connections

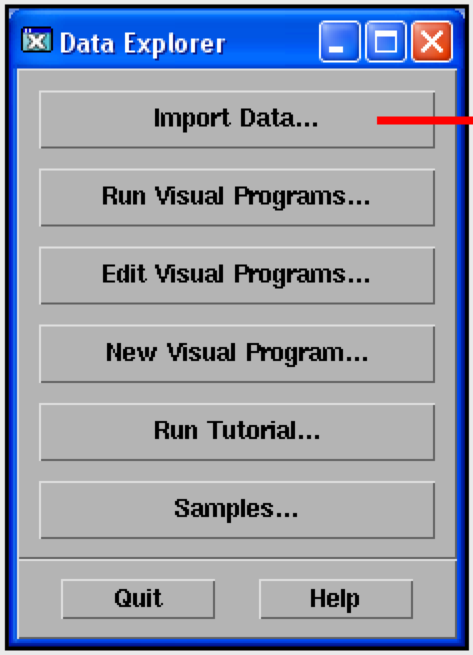
Deformed Regular:
IR positions,
R connections

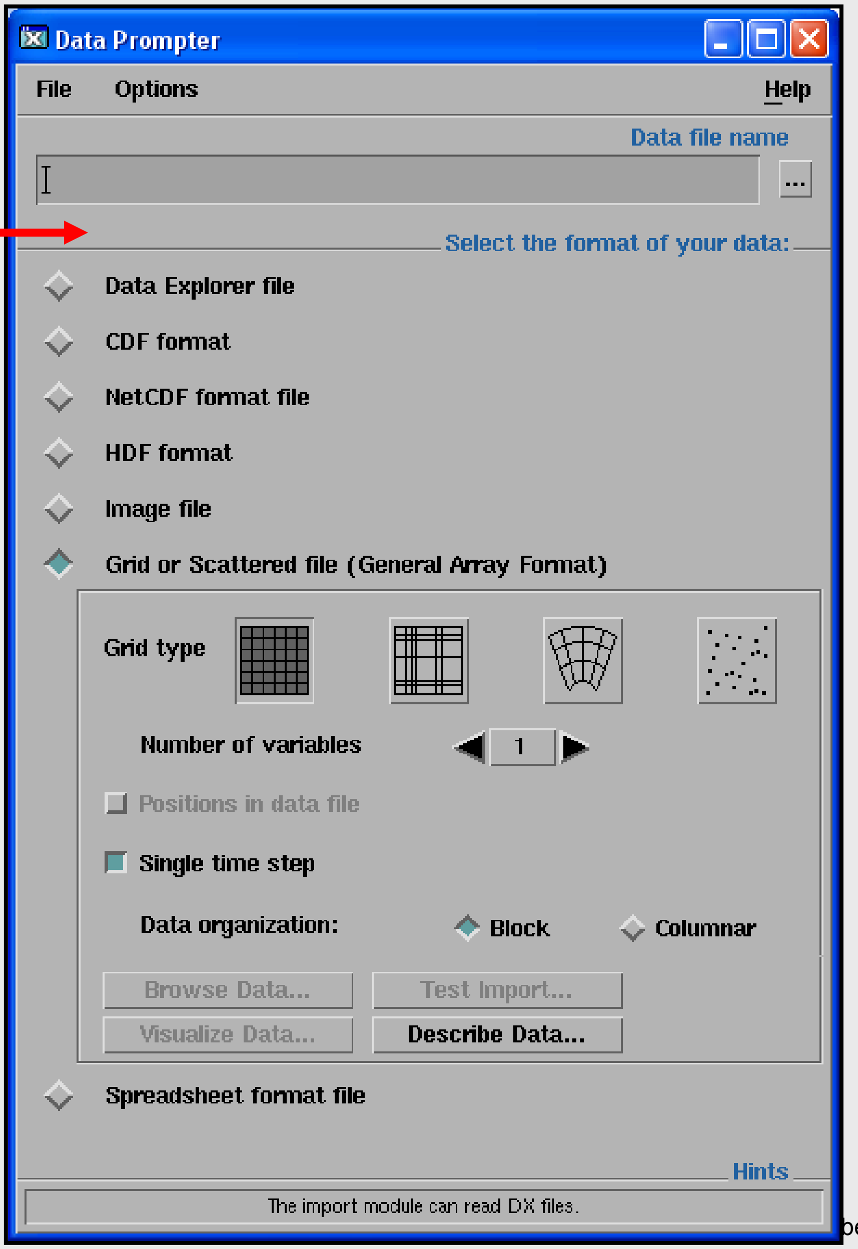
Irregular:
IR positions,
IR connections

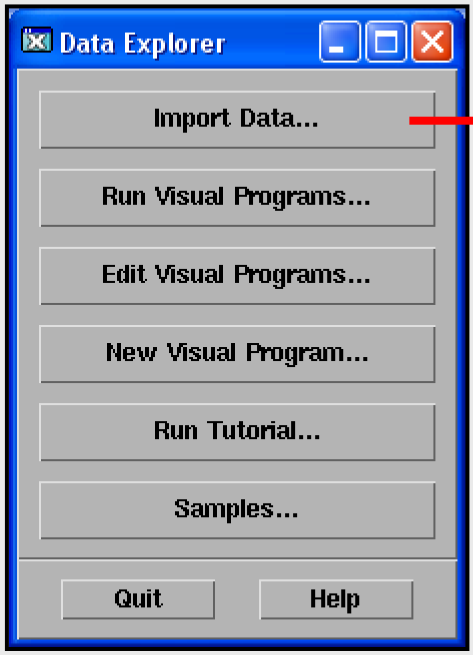
R = “regular”
IR = “irregular”

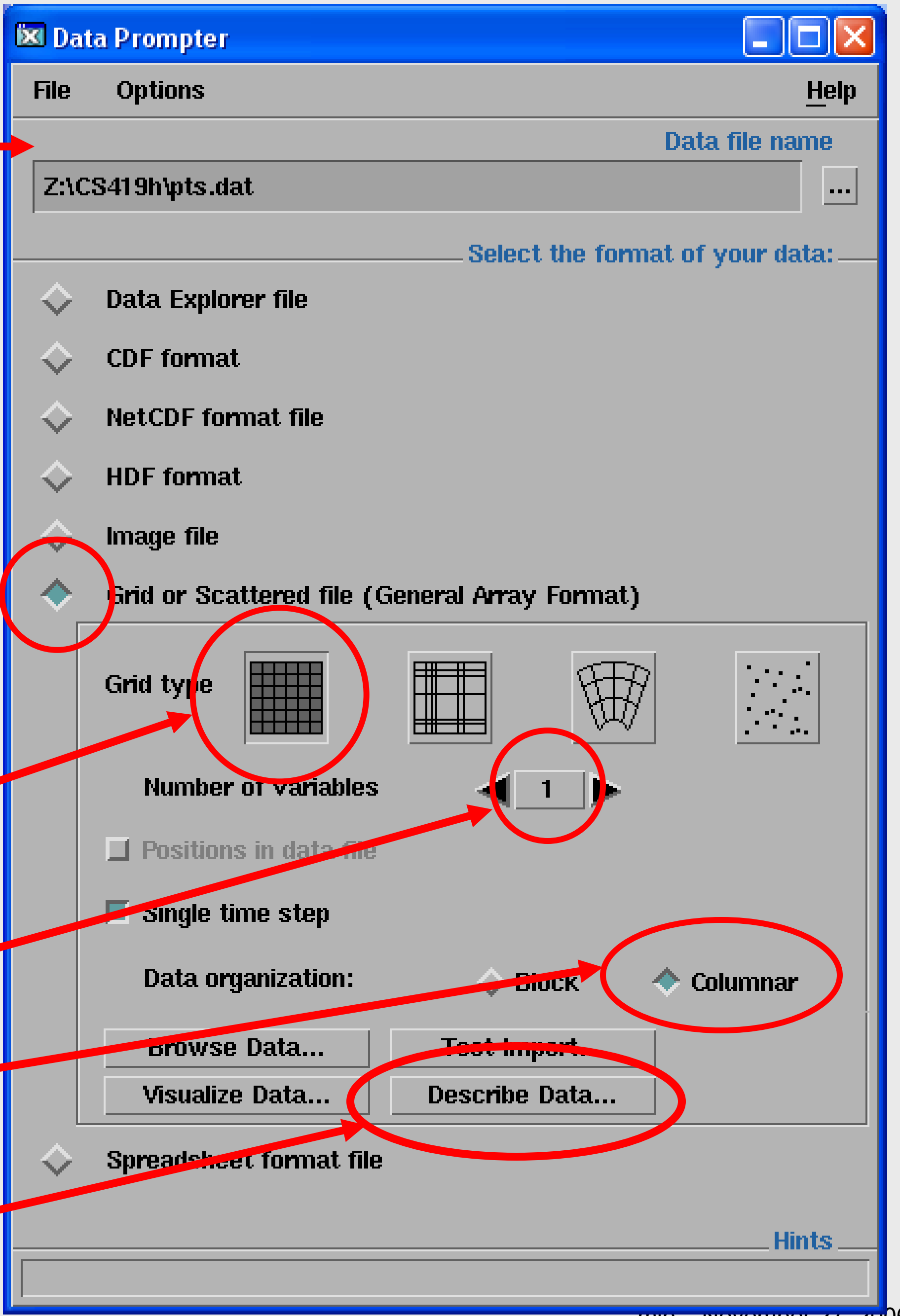
Oregon State University

Creating an OpenDX Data Descriptor File using the Data Prompter









Data file name: Z:\CS419h\pts.dat

Select the format of your data:

- Data Explorer file
- CDF format
- NetCDF format file
- HDF format
- Image file
- [✓ Grid or Scattered file (General Array Format)]
- Spreadsheet format file

Grid type

Number of variables: 1

Positions in data file

Single time step

Data organization: Block [✓ Columnar]

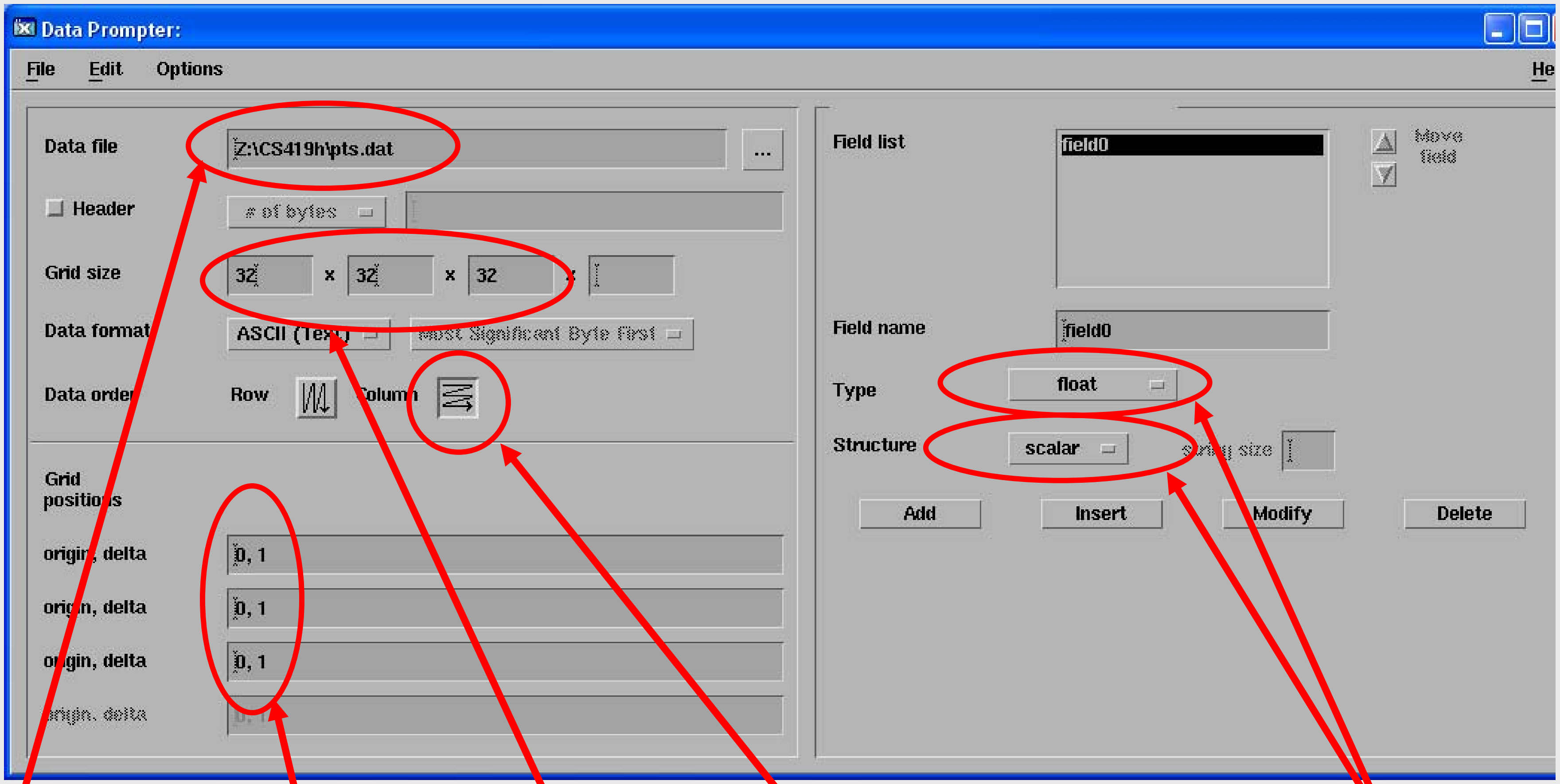
Most of the time, this is what you want

Regular positions, regular connections

One scalar value at each point

Doesn't matter here, but a good value for other data

See next page


Data Prompter:

File Edit Options He

Data file	Z:\CS419h\pts.dat
Header	# of bytes
Grid size	32 x 32 x 32 x
Data format	ASCII (Text) Most Significant Byte First
Data order	Row Column
Grid positions	
origin, delta	0, 1
origin, delta	0, 1
origin, delta	0, 1
origin, delta	0, 1

Field list	field0
Field name	field0
Type	float
Structure	scalar

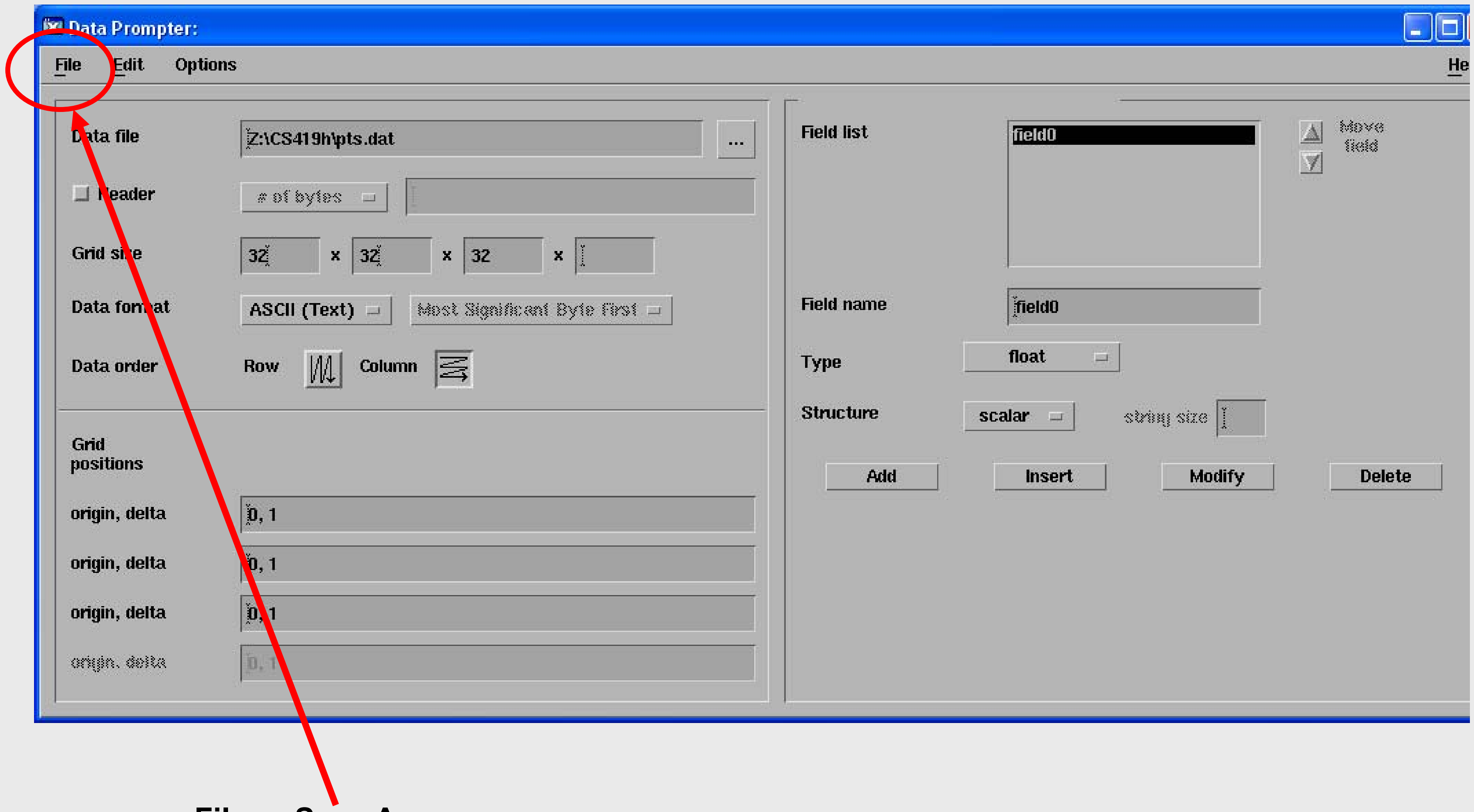
Enter the data filename

These get filled in for you when you enter the data dimensions – change them if you want

Enter the spatial dimensions

Doesn't matter here, but a good value for other data

Describe the data at each point (type and data dimension)



Data Prompter:

File Edit Options He

Data file: Z:\CS419h\pts.dat

Header: # of bytes

Grid size: 32 x 32 x 32 x

Data format: ASCII (Text) — Most Significant Byte First

Data order: Row / Column

Grid positions

origin, delta: 0, 1

origin, delta: 0, 1

origin, delta: 0, 1

origin, delta: 0, 1

Field list: field0

Move field

Field name: field0

Type: float

Structure: scalar — string size

Add Insert Modify Delete

File → Save As …

This saves the .general file, which will eventually tell OpenDX where to find the data and how to handle it.

The OpenDX .general File for a 3D Scalar Dataset

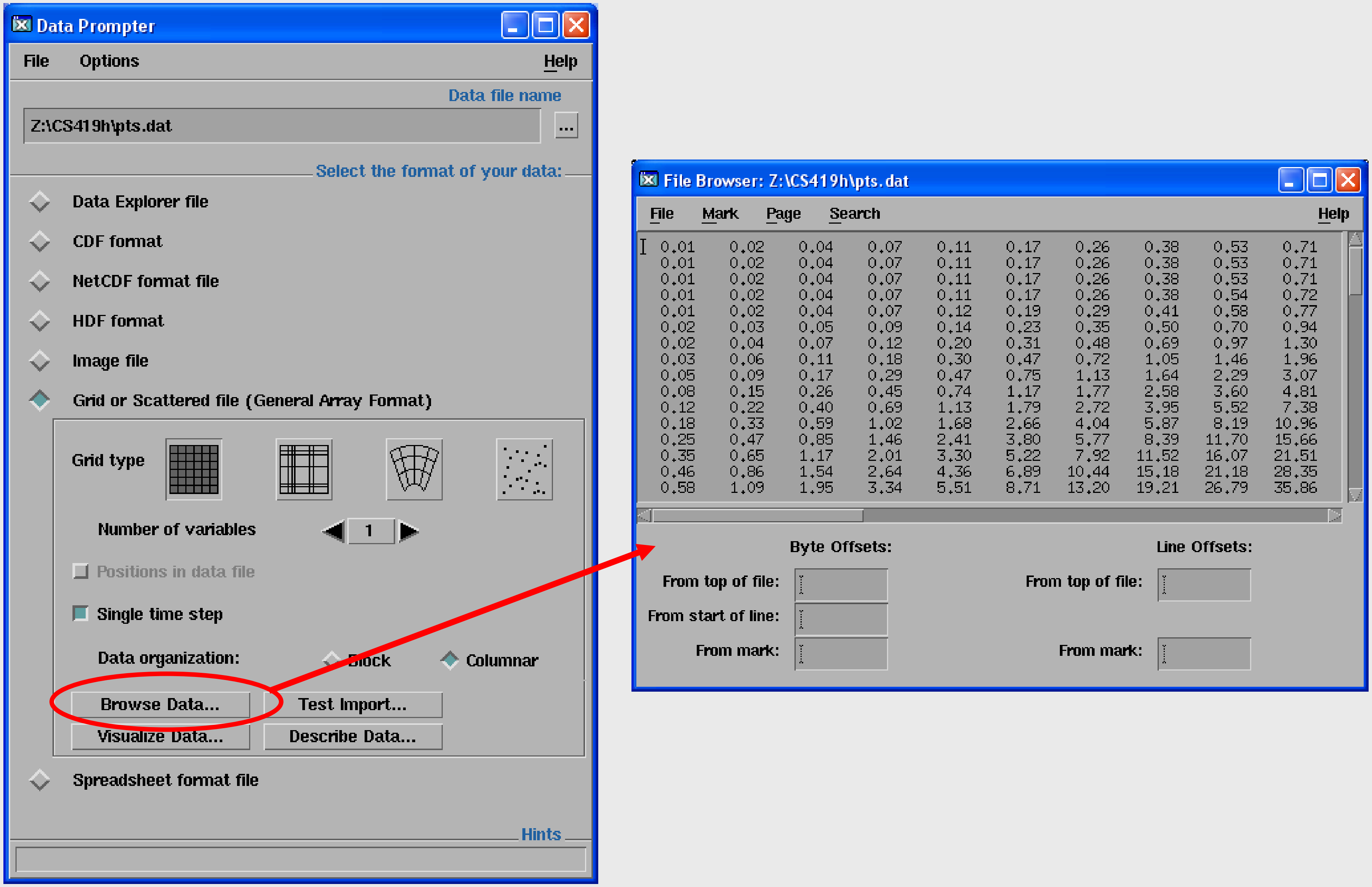
```
file = Z:\CS419h\pts.dat
grid = 32 x 32 x 32
format = ascii
interleaving = field
majority = row
field = field0
structure = scalar
type = float
dependency = positions
positions = regular, regular, regular, 0, 1, 0, 1, 0, 1

end
```

3D

Scalar

Regular positions

Oregon State University



Z:\CS419h\pts.dat

Data Explorer file
CDF format
NetCDF format file
HDF format
Image file
[✓ Grid or Scattered file (General Array Format)]

Grid type

Number of variables: 1

Positions in data file

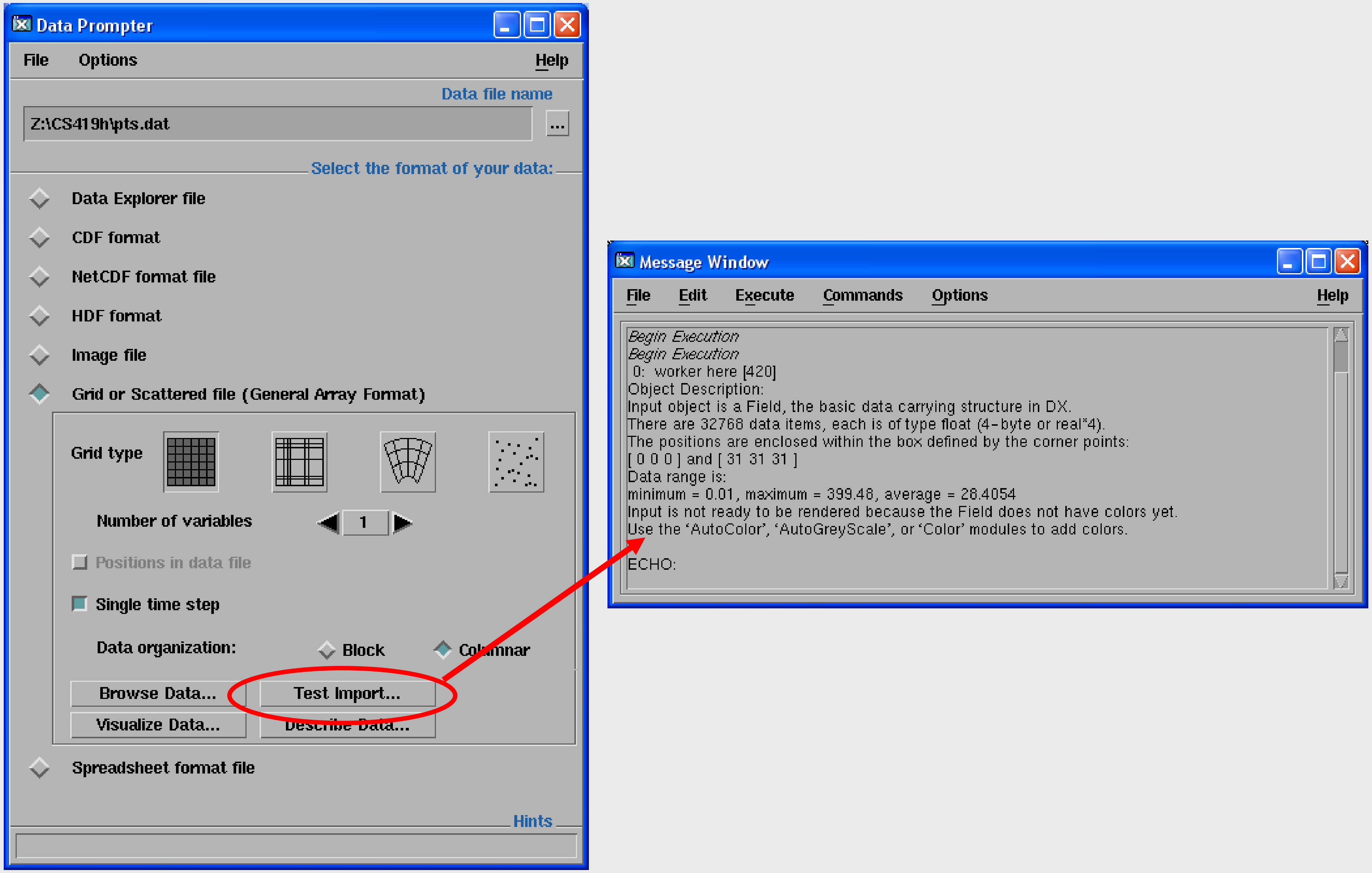
[✓ Single time step]

Data organization: Block [✓ Columnar]

Spreadsheet format file

```
 0.01   0.02   0.04   0.07   0.11   0.17   0.26   0.38   0.53   0.71
 0.01   0.02   0.04   0.07   0.11   0.17   0.26   0.38   0.53   0.71
 0.01   0.02   0.04   0.07   0.11   0.17   0.26   0.38   0.53   0.71
 0.01   0.02   0.04   0.07   0.11   0.17   0.26   0.38   0.54   0.72
 0.01   0.02   0.04   0.07   0.12   0.19   0.29   0.41   0.58   0.77
 0.02   0.03   0.05   0.09   0.14   0.23   0.35   0.50   0.70   0.94
 0.02   0.04   0.07   0.12   0.20   0.31   0.48   0.69   0.97   1.30
 0.03   0.06   0.11   0.18   0.30   0.47   0.72   1.05   1.46   1.96
 0.05   0.09   0.17   0.29   0.47   0.75   1.13   1.64   2.29   3.07
 0.08   0.15   0.26   0.45   0.74   1.17   1.77   2.58   3.60   4.81
 0.12   0.22   0.40   0.69   1.13   1.79   2.72   3.95   5.52   7.38
 0.18   0.33   0.59   1.02   1.68   2.66   4.04   5.87   8.19  10.96
 0.25   0.47   0.85   1.46   2.41   3.80   5.77   8.39  11.70  15.66
 0.35   0.65   1.17   2.01   3.30   5.22   7.92  11.52  16.07  21.51
 0.46   0.86   1.54   2.64   4.36   6.89  10.44  15.18  21.18  28.35
 0.58   1.09   1.95   3.34   5.51   8.71  13.20  19.21  26.79  35.86
```

Byte Offsets:		Line Offsets:	
From top of file:		From top of file:	
From start of line:			
From mark:		From mark:	

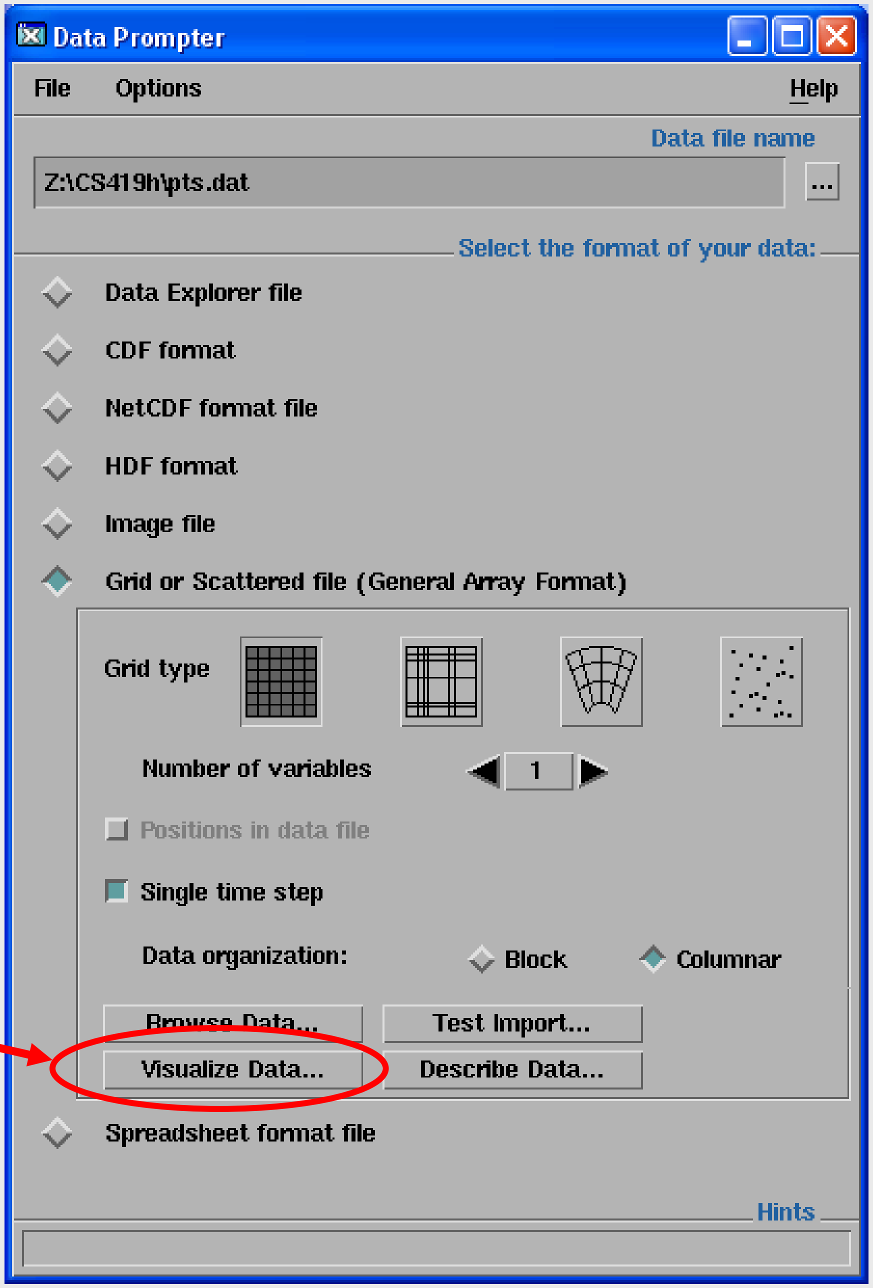


Data file name: Z:\CS419h\pts.dat

Select the format of your data:

- Data Explorer file
- CDF format
- NetCDF format file
- HDF format
- Image file
- [✓ Grid or Scattered file (General Array Format)]
  - Grid type
  - Number of variables: 1
  - Positions in data file
  - [✓ Single time step]
  - Data organization: Block [✓ Columnar]
- Spreadsheet format file

```
Begin Execution
Begin Execution
 0:  worker here [420]
Object Description:
Input object is a Field, the basic data carrying structure in DX.
There are 32768 data items, each is of type float (4-byte or real*4).
The positions are enclosed within the box defined by the corner points:
[ 0 0 0 ] and [ 31 31 31 ]
Data range is:
minimum = 0.01, maximum = 399.48, average = 28.4054
Input is not ready to be rendered because the Field does not have colors yet.
Use the 'AutoColor', 'AutoGreyScale', or 'Color' modules to add colors.

ECHO:
```



Data Prompter

File Options Help

Data file name: Z:\CS419h\pts.dat

Select the format of your data:

- Data Explorer file
- CDF format
- NetCDF format file
- HDF format
- Image file
- [✓ Grid or Scattered file (General Array Format)]

Grid type

Number of variables: 1

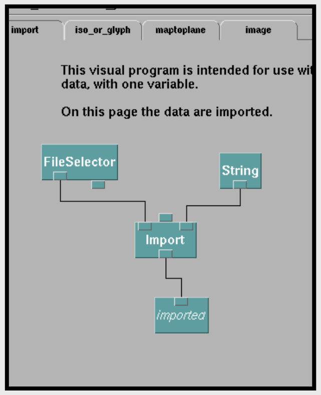
Positions in data file

[✓ Single time step]

Data organization: Block [✓ Columnar]

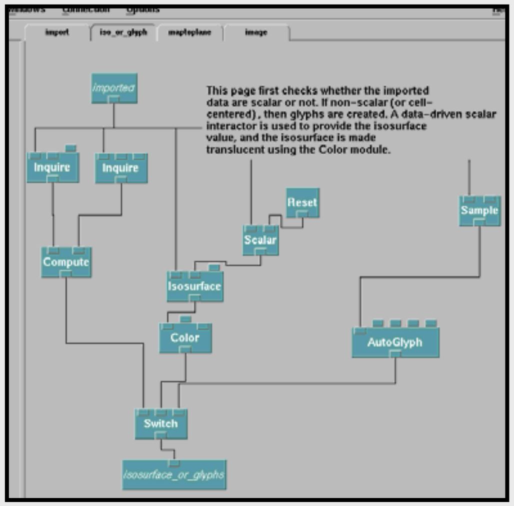
See next page

Oregon State University

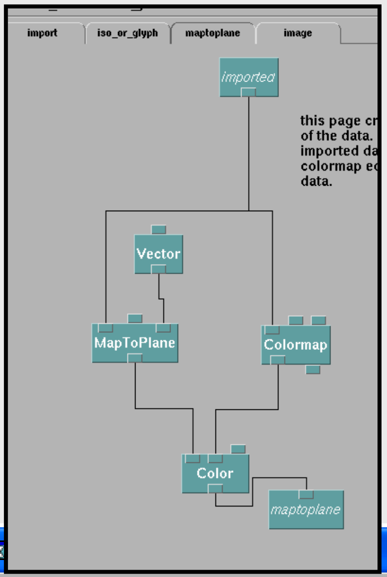


This visual program is intended for use wit
data, with one variable.

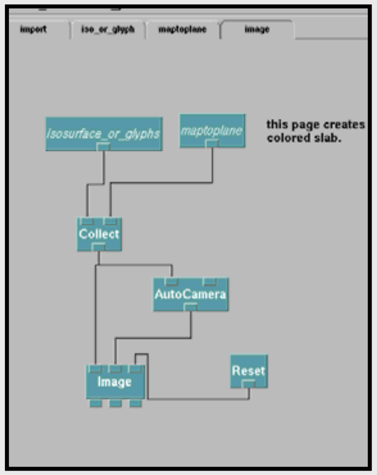
On this page the data are imported.



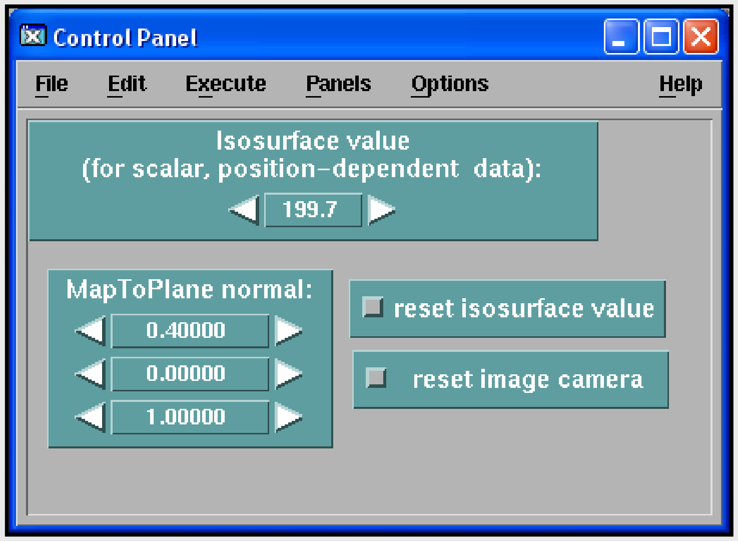
This page first checks whether the imported data are scalar or not. If non-scalar (or cell-centered), then glyphs are created. A data-driven scalar interactor is used to provide the isosurface value, and the isosurface is made translucent using the Color module.



this page cr
of the data.
imported da
colormap ec
data.

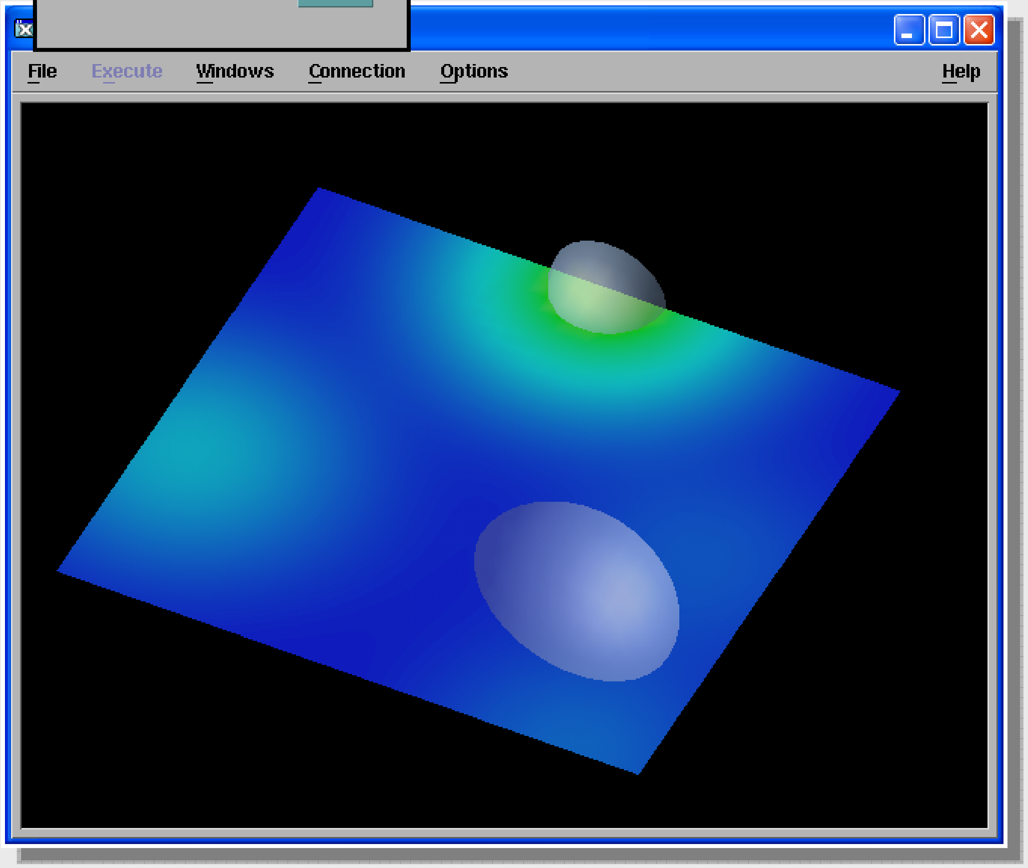


this page creates
colored slab.



Isosurface value
(for scalar, position-dependent data):
199.7

MapToPlane normal:
0.40000
0.00000
1.00000



This all gets created automatically!
Pretty amazing, huh?

Oregon State University

Terrain Visualization .general File

```
file = Z:\OpenDX\or.dat
grid = 201 x 105
format = ascii
interleaving = record
majority = column
field = field0
structure = scalar
type = float
dependency = positions
positions = regular, regular, 0, 1, 0, -1

end
```

2D

Scalar

Regular positions

Vector Field Visualization .general File

```
file = Z:\OpenDX\vecs.dat
grid = 16 x 16 x 16
format = ascii
interleaving = record-vector
majority = column
field = velocity
structure = 3-vector
type = float
dependency = positions
positions = regular, regular, regular, 0, 1, 0, 1, 0, 1

end
```

3D

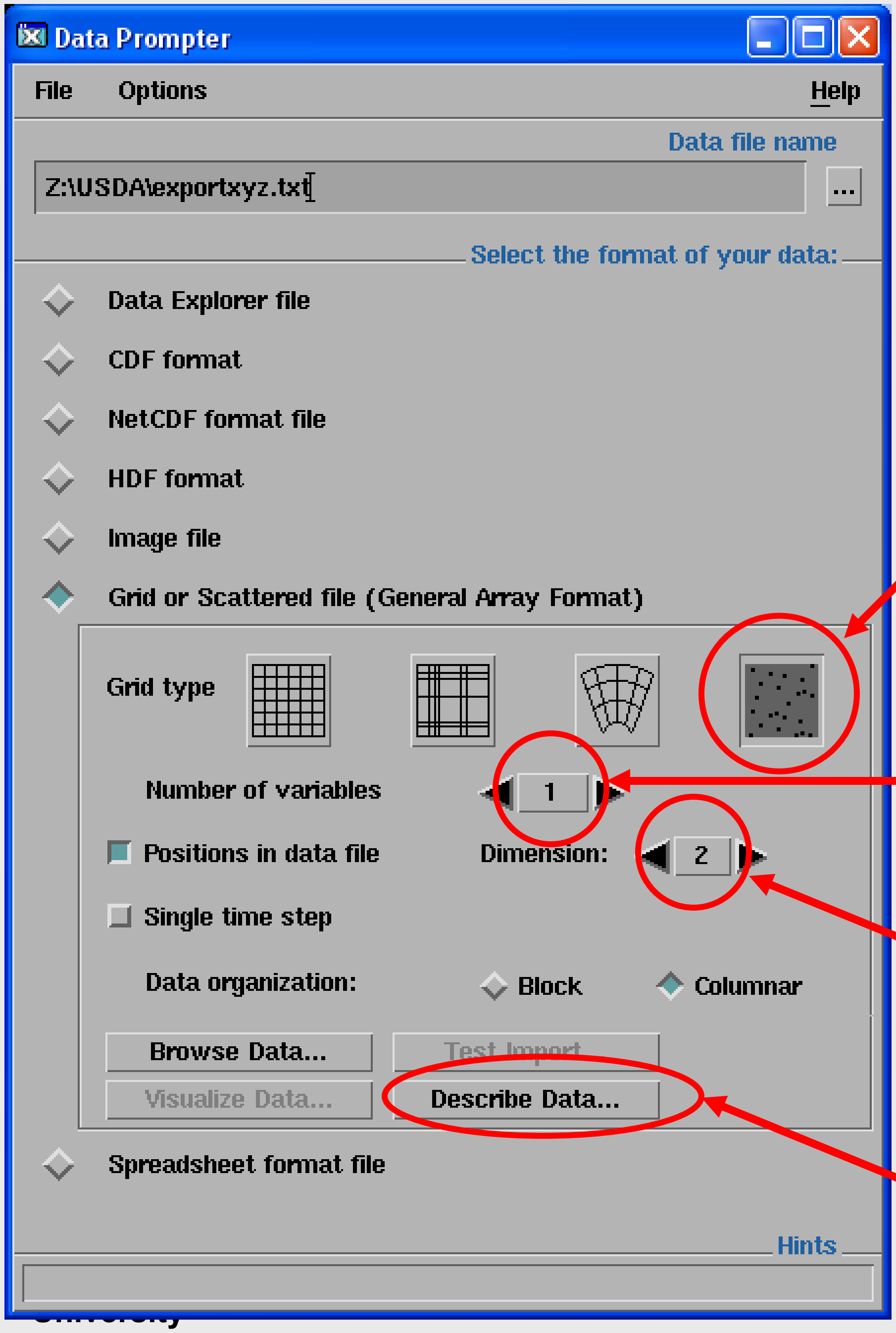
3-element Vector

Regular positions

Oregon State University

Visualizing Points on a Scattered Grid (e.g., Digital Elevation Mapping)

West→East	South→North	Elevation
-360.78	128.438	2.25
-360.75	128.428	2.31
-360.80	128.405	2.20
-360.81	128.370	1.99
-360.91	128.369	1.75
-361.00	128.359	1.65
-361.16	128.354	1.77
-361.21	128.344	1.70
-361.25	128.344	1.76
. . .		



Data Prompter

File Options Help

Data file name: Z:\USDA\exportxyz.txt

Select the format of your data:

- Data Explorer file
- CDF format
- NetCDF format file
- HDF format
- Image file
- [✓ Grid or Scattered file (General Array Format)]
- Spreadsheet format file

Grid type: [✓ scattered]

Number of variables: 1

[✓ Positions in data file] Dimension: 2

Single time step

Data organization: Block [✓ Columnar]

Browse Data... Test Import

Visualize Data... Describe Data...

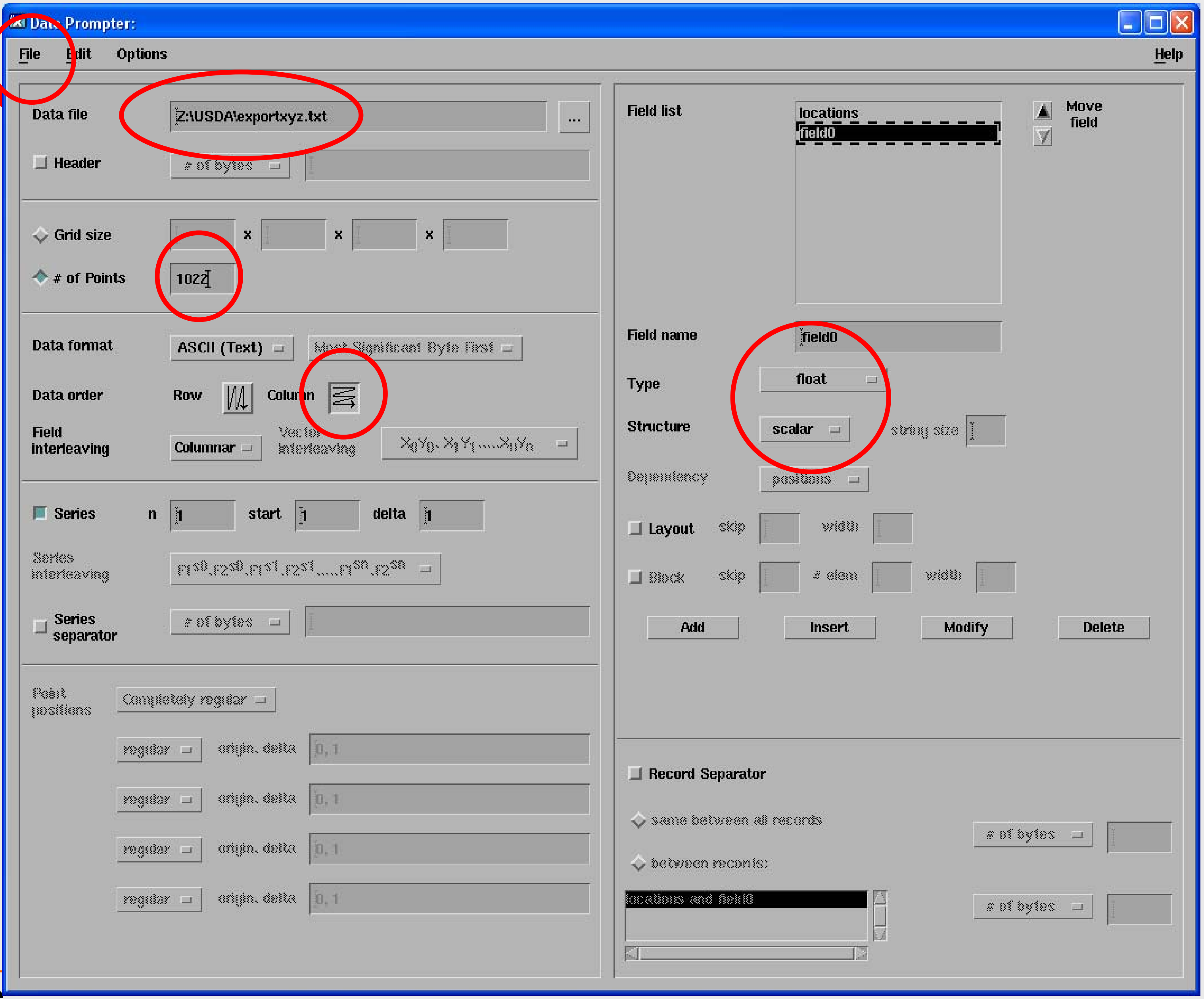
Hints

Scattered Data (irregular positions, irregular connections)

Data Dimension: a scalar value at each position

Spatial Dimension: 2D

See next page


Data Prompter:

File Edit Options Help

Data file: Z:\USDA\exportxyz.txt

Header: # of bytes

Grid size: x x x

# of Points: 1022

Data format: ASCII (Text) — Most Significant Byte First

Data order: Row / Column

Field interleaving: Columnar — Vector interleaving: $X_0Y_0, X_1Y_1, \ldots, X_nY_n$

Series: n 1 start 1 delta 1

Series interleaving: $F1^{S0}, F2^{S0}, F1^{S1}, F2^{S1}, \ldots, F1^{Sn}, F2^{Sn}$

Series separator: # of bytes

Point positions: Completely regular

regular origin, delta 0, 1

regular origin, delta 0, 1

regular origin, delta 0, 1

regular origin, delta 0, 1

Field list: locations, field0

Move field

Field name: field0

Type: float

Structure: scalar — string size

Dependency: positions

Layout: skip, width

Block: skip, # elem, width

Add Insert Modify Delete

Record Separator

same between all records: # of bytes

between records: # of bytes

locations and field0

File → Save as…

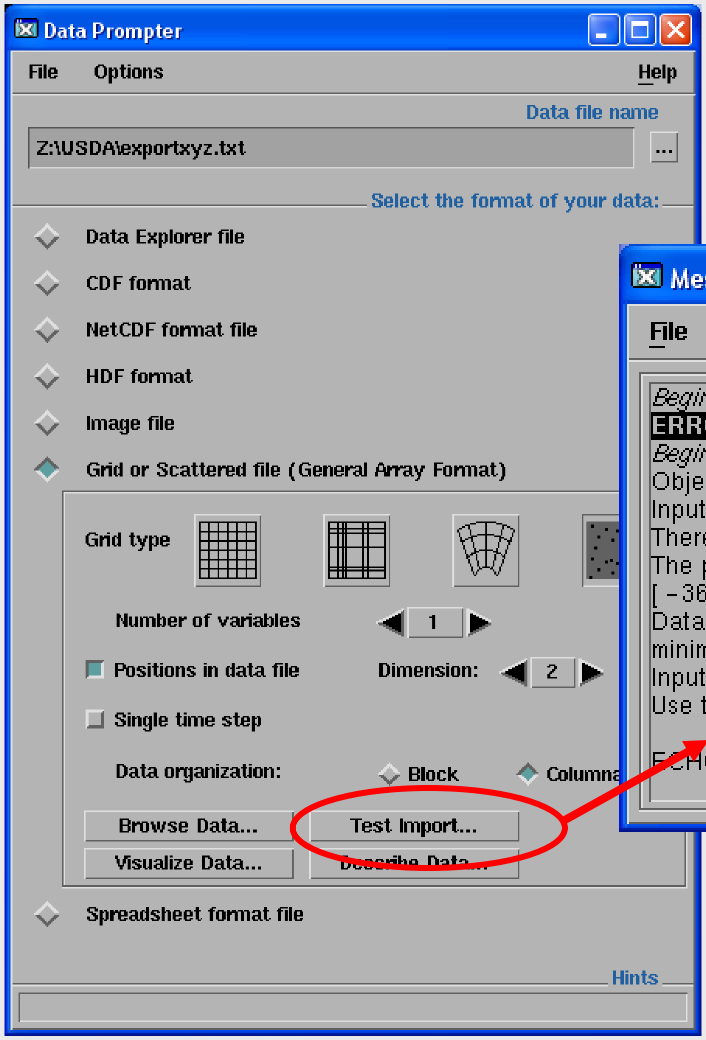
A Scattered Grid .general File

```
file = Z:\OpenDX\exportxyz.dat
points = 1022
format = ascii
interleaving = field
field = locations, field0
structure = 2-vector, scalar
type = float, float

end
```

A 1D list

Each entry has a location and data values

Each location has 2 values (X,Y). Each data is a single value (scalar).


Data Prompter

File Options Help

Data file name: Z:\USDA\exportxyz.txt

Select the format of your data:

- Data Explorer file
- CDF format
- NetCDF format file
- HDF format
- Image file
- [✓ Grid or Scattered file (General Array Format)]

Grid type

Number of variables: 1

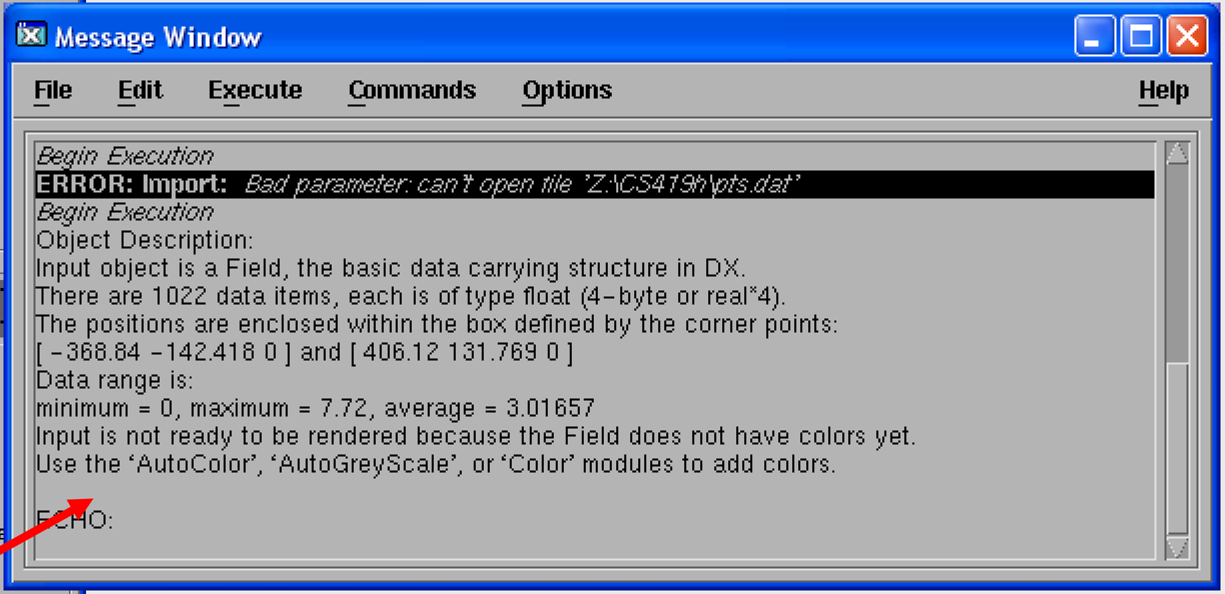
[✓ Positions in data file] Dimension: 2

Single time step

Data organization: Block [✓ Columnar]

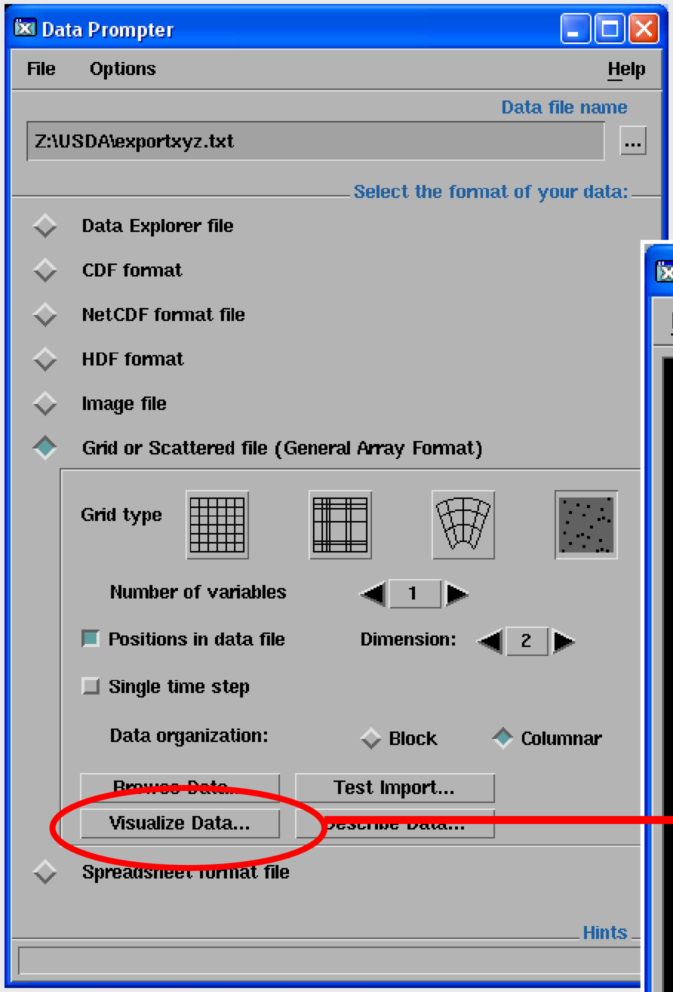
- Spreadsheet format file

Hints



```
Begin Execution
ERROR: Import:  Bad parameter: can't open file 'Z:\CS419h\pts.dat'
Begin Execution
Object Description:
Input object is a Field, the basic data carrying structure in DX.
There are 1022 data items, each is of type float (4-byte or real*4).
The positions are enclosed within the box defined by the corner points:
[ -368.84 -142.418 0 ] and [ 406.12 131.769 0 ]
Data range is:
minimum = 0, maximum = 7.72, average = 3.01657
Input is not ready to be rendered because the Field does not have colors yet.
Use the 'AutoColor', 'AutoGreyScale', or 'Color' modules to add colors.

ECHO:
```

OSU Oregon State University



Data Prompter

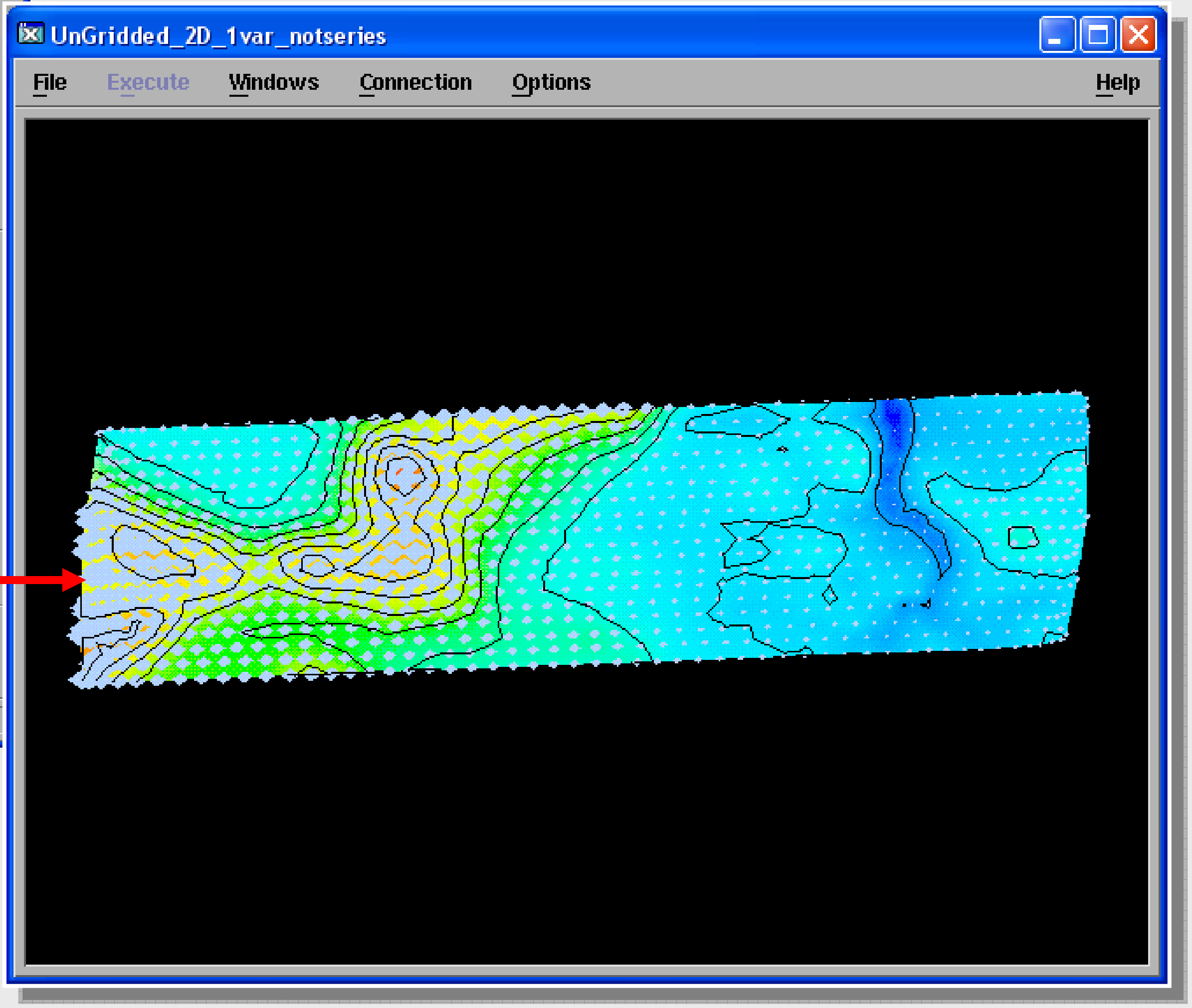
File Options Help

Data file name: Z:\USDA\exportxyz.txt

Select the format of your data:

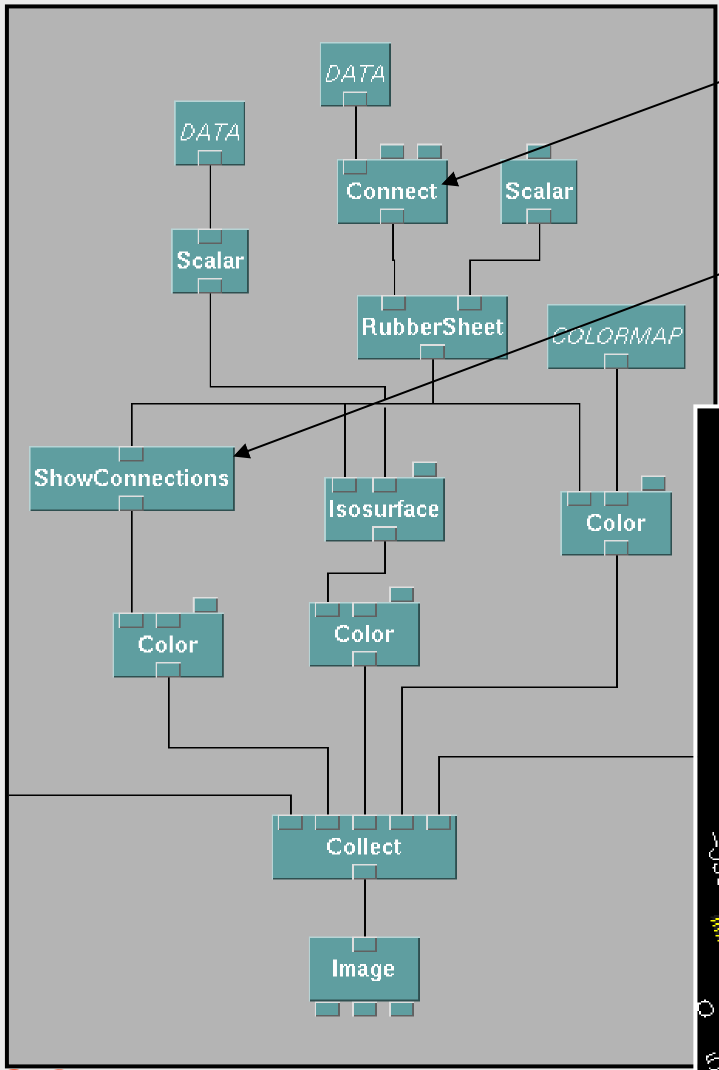
- Data Explorer file
- CDF format
- NetCDF format file
- HDF format
- Image file
- Grid or Scattered file (General Array Format) [✓]
  - Grid type
  - Number of variables: 1
  - [✓] Positions in data file — Dimension: 2
  - [ ] Single time step
  - Data organization: Block [✓ Columnar]
  - Browse Data... | Test Import... | Visualize Data... | Describe Data...
- Spreadsheet format file

Hints


UnGridded_2D_1var_notseries
File Execute Windows Connection Options Help

Oregon State University

Visualizing Points from a Scattered Grid (e.g., Digital Elevation Mapping)

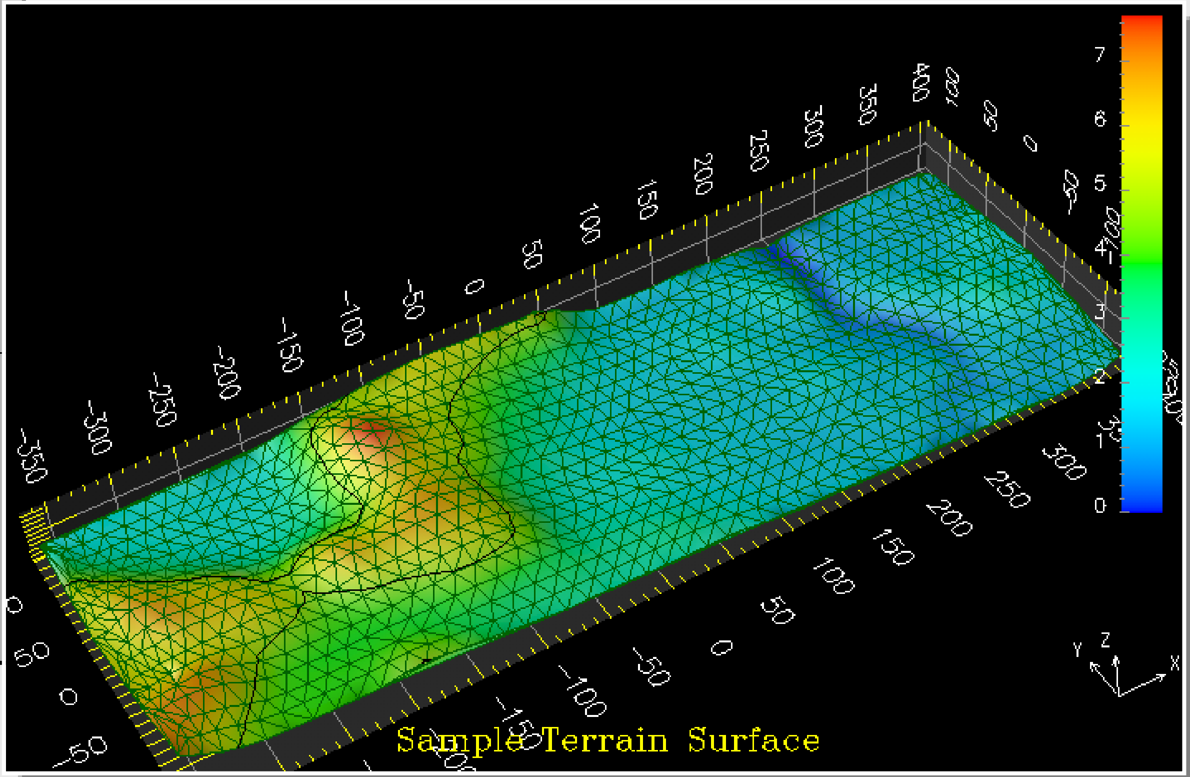


Realization

Realization

Connect creates triangles from scattered points.

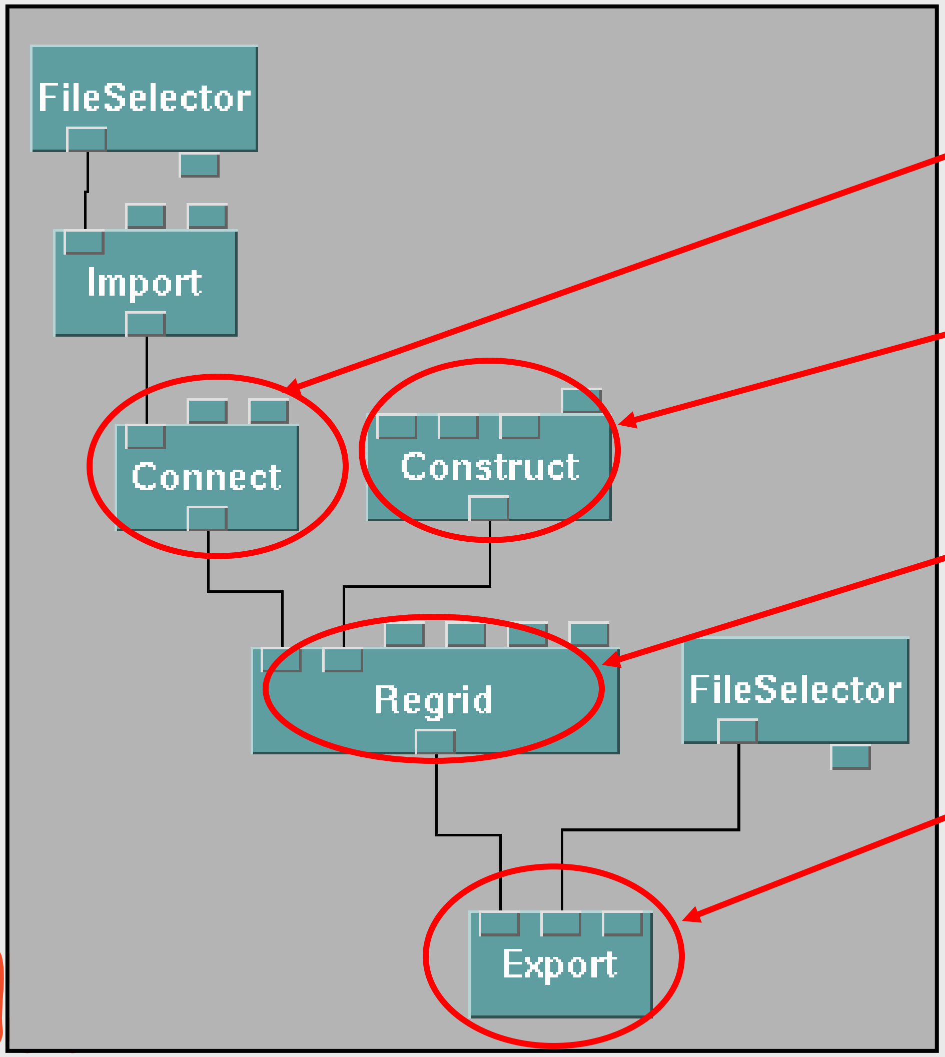
ShowConnections creates graphics to show how the scattered point connections were made.



Oregon State University

Regridding a Dataset
(especially good for writing out a Connect'ed Scattered Grid as a Rectangular Array)

You might be doing this to downsize a dataset or to create a regular grid from a scattered grid



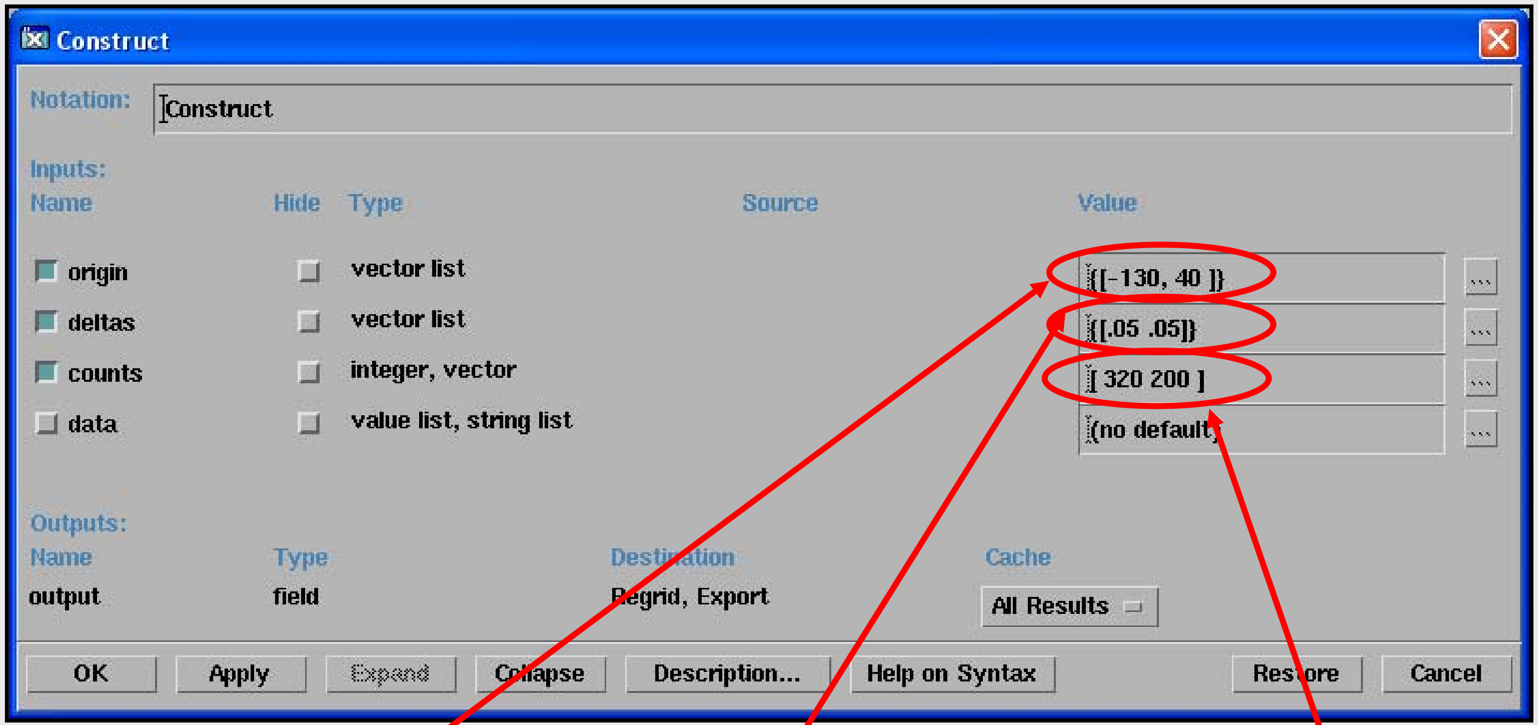
If Scattered data, route through Connect first.

Creates a new grid to project the data onto

Projects the original data onto the new grid

Writes out the new grid data file

Regridding a Dataset



Construct

Notation: Construct

Inputs:

Name	Hide	Type	Source	Value
origin		vector list		{[-130, 40]}
deltas		vector list		{[.05 .05]}
counts		integer, vector		[320 200]
data		value list, string list		(no default)

Outputs:

Name	Type	Destination	Cache
output	field	Regrid, Export	All Results

Coordinates at the lower corner of the dataset

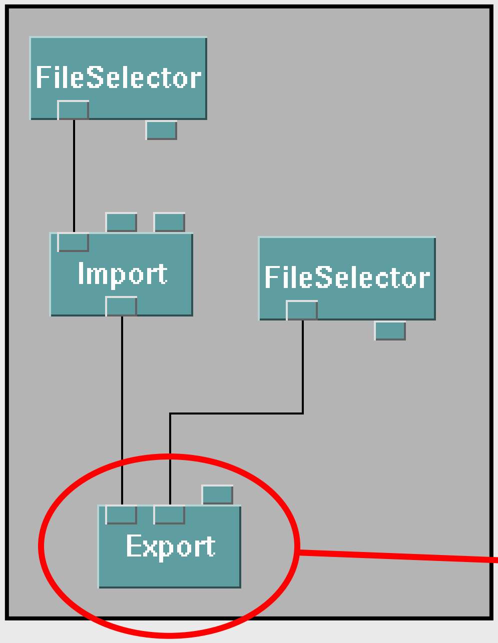
Step size in each dimension

How many steps to make in the grid

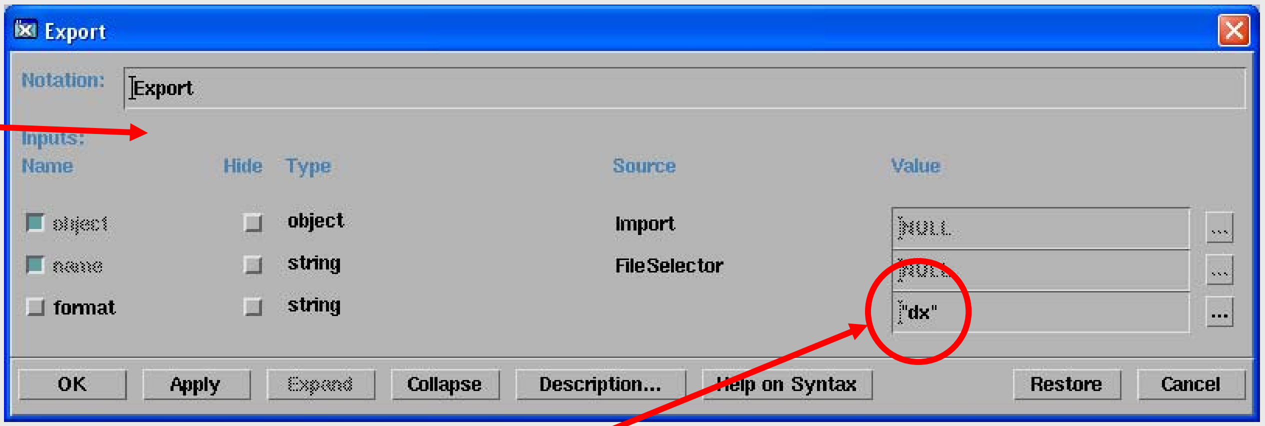
Note: the upper corner coordinates will be "origin + (counts-1)*delta"

Oregon State University

Writing a .general Dataset as a Native OpenDX .dx Dataset



The .dx file format embeds the data description information, the positions, and the data values in one file. This makes it easier to keep track of and easier to give to other people.



Notation: Export

Inputs:

Name	Hide	Type	Source	Value
object		object	Import	NULL
name		string	FileSelector	NULL
format		string		"dx"

Says that you want to Export the data in a native OpenDX .dx file form.

Oregon State University

Send comments and suggestions on these notes to:

Mike Bailey
Professor, Computer Science
Oregon State University
2117 Kelley Engineering Center
Corvallis, OR 97331-5501
541-737-2542
FAX: 541-737-1300
mjb@cs.oregonstate.edu
http://eecs.oregonstate.edu/~mjb