OpenDX, abbreviated for CS 553

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://cs.oregonstate.edu/~mjb/opendx
The Structure of an OpenDX Module

Inputs

Code that does something useful

Outputs

OpenDX Module
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
The OpenDX Main Menu and Categories of Modules
The OpenDX Main Menu

- Get into the Data Prompter program
- Run an OpenDX network and be able to edit the network
- Run the internal OpenDX tutorial
- Exit OpenDX

- Run an OpenDX network without seeing the network
- Create a new OpenDX network
- Load, and be able to edit, one of the OpenDX sample networks
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 specifying 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 or 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

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

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

Collect and Compute are two common modules that work this way.
Terrain Visualization
Terrain Visualization

Import/Export

Transformation

Rendering

Start simply, then embellish!
You can type a filename into the Import module, but hooking in a FileSelector module makes it way easier and friendlier.
The Colormap Editor Module

- 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

Oregon State University
Computer Graphics
Rubbersheeting the Terrain Surface

- Import
- Colormap
- Color
- RubberSheet
- Collect
- Image
- Scalar
- Isosurface
- Color

[Diagram of terrain surface processing flow]
The Image Window
The *AutoAxes* option has many ways to embellish the visualization with axes, labels, grids, etc.
The *Mode* option lets you set what scene transformation the mouse will perform.
The View Control option lets you set various aspects of how the scene will appear.

**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.
Scalar Visualization
Glyphs

Import

Scalar

AutoGlyph

Colormap

Color

Collect

Image

Glyph scale: 2.70
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.
3D Cutting Plane – Contours
Isosurfaces

Realization

Opaque Isovalue: 99.3
Translucent Isovalue: 55.9
Isosurface Opacity: 0.25
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”
Vector Visualization
Vector Cloud

OSU
Oregon State University
Computer Graphics

mjb – March 6, 2009
Speed Isosurfaces

Compute

Expression:
\[ \sqrt{a.x^2 + a.y^2 + a.z^2} \]
Streamline Ribbon

Realization

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

Annotation

Expression:
\[ \sqrt{a \cdot x^2 + a \cdot y^2 + a \cdot z^2} \]
Divergence

Transformation