IEEE Visualization Tutorial
Hank Childs
October 10, 2004

The VisIt Team:
Eric Brugger (project leader),
Kathleen Bonnell, Hank Childs, Jeremy Meredith,
Mark Miller, and Brad Whitlock
Alum: Sean Ahern

Gas bubble subjected to shock (Raptor)

UCRL-PRES -207039
Work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract W-7405-Eng-48.
VisIt: Background

• Background
• Design
• Big data handling
• Extensibility & Future Directions

• What space is VisIt filling?
• What types of data?
• What types of operations does VisIt support?
• What is the target environment?
• What is the availability of VisIt?
What space is VisIt filling?

• VisIt is an end user tool.
  – Used by: physicists, engineers, code developers, vis experts.
  – Goals: robustness, usability, performance
• Emphasis on large data and unusual data models
• VisIt is used for:

  - Data Exploration
  - Data Analysis
  - Debugging Simulation Codes
  - Presentations
What types of data can VisIt handle?

- **Meshes:**
  - 2D and 3D unstructured meshes, curvilinear, rectilinear, point meshes, and AMR meshes.

- **Variables:**
  - Scalar, vector, material, species, tensor

- **Selection:**
  - Domain, group, arbitrary
What types of operations does VisIt support?

<table>
<thead>
<tr>
<th>Data Exploration</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Slicing by plane, sphere, cone…</td>
</tr>
<tr>
<td>• Contouring</td>
</tr>
<tr>
<td>• Volume rendering</td>
</tr>
<tr>
<td>• Alter zones displayed (clip, …)</td>
</tr>
<tr>
<td>• Expressions to create new variables (a+b, gradient(a), etc)</td>
</tr>
<tr>
<td>• Much more…</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Produce 1D curve along line based on scalar value (“lineout”)</td>
</tr>
<tr>
<td>• Ops on 1D curves (L2Norm, …)</td>
</tr>
<tr>
<td>• Surface area, volume, etc.</td>
</tr>
<tr>
<td>• All queries can be taken over all timesteps to produce 1D curve.</td>
</tr>
<tr>
<td>• Much more…</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Debug Simulation Codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Query per-zone values by picking on screen</td>
</tr>
<tr>
<td>• Show only zones that meet some criteria (ie high temperature zones/ hot spots)</td>
</tr>
<tr>
<td>• Much more…</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Presentations</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Controls over much of the final pictures look and feel – annotations, colors, etc.</td>
</tr>
<tr>
<td>• Complete movie creation, including final MPEG generation.</td>
</tr>
<tr>
<td>• Much more…</td>
</tr>
</tbody>
</table>
What is the target environment for VisIt?

- Parallel vis machine w/ access to data
  - Could be resource rich or poor
- Desktop machines with good graphics cards
- Gigabit interconnect

➢ This is the target environment for VisIt. This environment is not necessary for the tool to run successfully.

parallel machine – eg ASCI white

Blue Gene/L, 65536 procs., # vis procs to be determined (<512).

ASC white, 8192 procs., 512 vis procs.
What is the availability of VisIt?

- Publicly available (http://www.llnl.gov/visit).
- Supported by 6 LLNL developers.
- Support for Linux, AIX, Tru64, Solaris, IRIX, Windows, and Mac.
VisIt: Design

- Background
- **Design**
- Big data handling
- Extensibility & Future Directions

- Basic architecture
- Data flow network design
Basic Architecture

- Distributed design to leverage parallel compute resources and graphics hardware.
- Three major components – “UI”, “Viewer”, “Engine”

**UI responsible for user interface**
- Designed to easily add new UI components
- Currently
  - Qt-based graphical
  - Python-based CLI

**Viewer responsible for:**
- Windowing and rendering
- Managing centralized state
  - Enables fault tolerance

**Engine responsible for all data management**
- Parallelized
- Based on data flow networks
Data Flow Networks

- Two primary types: data objects and process objects
- Three basic types of process objects: source, sink, filter

C++-inheritance. The Silo reader “is a” source.

Key

<table>
<thead>
<tr>
<th>Abstract Type</th>
<th>Concrete Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>Filter</td>
</tr>
<tr>
<td>Silo Reader</td>
<td>Slice</td>
</tr>
<tr>
<td></td>
<td>Contour</td>
</tr>
<tr>
<td></td>
<td>Rigid-body Transform</td>
</tr>
<tr>
<td>Sink</td>
<td>Triangle rendering algorithm</td>
</tr>
</tbody>
</table>
Pipelines

• A pipeline is created with a source, sink, and many filters.

• Connections are made through data objects (output to input).

• Demand driven

• Two contracts for pipeline:
  – One describes what the input to a filter should look like
  – Other describes what the output of a filter actually has.
VisIt: Big data handling

- Background
- Design
- **Big data handling**
- Extensibility & Future Directions

- How does VisIt parallelize work?
- Rendering large surfaces
- Ghost data
- Removing unneeded data
How does VisIt parallelize work?

- Identical pipelines on each processor.
- Domain overloading
- All domains executed by Filter A before moving on to Filter B
  - Enables collective communication.
  - Streaming still possible, but not implemented.
Rendering large surfaces

- VisIt has two rendering modes:
  - Hardware accelerated
  - Scalable rendering mode
- Hardware accelerated
  - Brings triangles to Viewer and uses graphics card
- Scalable rendering
  - Leave triangles on Engine.
  - Do parallel rendering to get image.
  - Transfer image to local desktop
  - Must get new image for each view frustum.
Ghost data

- Using domain decomposition leads to artifacts along the domain boundaries.
- Solved by ghost data
  - Ghost zone and ghost nodes
  - Filters request what type of ghost data they need.

Poor interpolation along domain boundaries leads to discontinuous isosurfaces.

Faces external to a domain can be internal to the problem. Causes triangle bloat & transparency issues.

Ghost data solves these problems!
Removing Unneeded Data

- Not all data is necessary to make the “final picture”
  - Ex: Slice only affects some domains.
- Meta-data can be used to prevent data from ever being operated on.
  - Each filter can reduce set of domains considered.

Consult per-domain spatial extents & remove domains that don’t intersect slice.
VisIt

- Background
- Design
- Big data handling
- Extensibility & Future Directions
Extensibility: Plugins

- VisIt has a plugin architecture.
  - Databases, operators, and plots
- May be more plugins in the future
  - Expressions? Queries?

- Fully code-generated:
  - Window
  - Attributes
  - Bindings to VisIt

- Partially code-generated:
  - Filter to do operation

```
"xml2plugin"
```

```
"xmledit"
```

```
OnionPeel.xml
```
Future Directions

• Connecting to a running simulation.
  – Alpha version up and going.

• Comparing databases.
  – Recent code to lock/correlate multiple databases in time.

• Allowing queries to be used anywhere in the code.
  – Threshold by Query(90th percentile, “pressure”), …
Future Directions: Continuing to make a usable tool for our user base.

- VisIt’s usage is high (~1000 startups per month).
- 13 releases in last 12 months
  - 800 enhancements/bugs resolved.
  - Majority of work is on robustness, usability, and performance.
  - Regression suite has grown to over 700 tests.
  - Extensive on-line and off-line documentation.

- 3 classes on VisIt were given.
  - 302 slides + >60 exercises.

Slides from the VisIt class