Visualization and Analysis of Massive Data with VisIt

ATPESC 2015
Argonne Training Program on Extreme-Scale Computing
Wednesday August 12, 2015

Cyrus Harrison
Lawrence Livermore National Laboratory
cyrush@llnl.gov
Tutorial Outline

- VisIt Project Intro [10 min]
- Guided tour of VisIt [20 min]
- Hands on with an Aneurysm Simulation [30 min]
Tutorial Resources

- Tutorial Prep:

- Example Datasets:
  http://visitusers.org/index.php?title=Tutorial_Data

- Blood Flow Hands-on:

- More Tutorial Materials (From past Tutorials):
  http://visitusers.org/index.php?title=VisIt_Tutorial

- Cyrus’ Email: cyrush@llnl.gov
Aneurysm Simulation Data

Simulated using the LifeV (http://www.lifev.org/) finite element solver.

Available thanks to:

Gilles Fourestey and Jean Favre
Swiss National Supercomputing Centre

http://www.cscs.ch/
VisIt Project Introduction
VisIt is an open source, turnkey application for data analysis and visualization of mesh-based data.

- Production end-user tool supporting scientific and engineering applications.
- Provides an infrastructure for parallel post-processing that scales from desktops to massive HPC clusters.
- Source released under a BSD style license.
VisIt supports a wide range of use cases.

- **Data Exploration**
- **Comparative Analysis**
- **Quantitative Analysis**
- **Visual Debugging**
- **Presentation Graphics**
Examples of VisIt’s visualization capabilities.
VisIt uses MPI for distributed-memory parallelism on HPC clusters.

Full Dataset (27 billion total cells)

3072 sub-grids (each 192x129x256 cells)

We are enhancing VisIt’s pipeline infrastructure to support threaded processing and many-core architectures.
VisIt is a vibrant project with many participants.

- The VisIt project started in 2000 to support LLNL’s large scale ASC physics codes.

- The project grew beyond LLNL and ASC with research and development from DOE SciDAC and other efforts.

- VisIt is now supported by multiple organizations:
  - LLNL, LBNL, ORNL, UC Davis, Univ of Utah, Intelligent Light, ...

- Over 75 person years of effort, 1.5+ million lines of code.
VisIt’s capabilities are constantly being expanded.

Ongoing + Recent Development Efforts:

- Evaluation of EAVL / VTK-M for batch In-situ processing
  - [http://m.vtk.org/index.php/Main_Page](http://m.vtk.org/index.php/Main_Page)
  - [http://ft.ornl.gov/eavl/](http://ft.ornl.gov/eavl/)

- Built-in SeedMe Python support for sharing visualizations
  - [http://seedme.org/](http://seedme.org/)

- Support for High Order Finite Element Meshes via MFEM
  - [https://code.google.com/p/mfem/](https://code.google.com/p/mfem/)
## Project Introduction

VisIt scales well on current HPC platforms.

<table>
<thead>
<tr>
<th>Machine</th>
<th>Architecture</th>
<th>Problem Size</th>
<th># of Cores</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graph</td>
<td>X86_64</td>
<td>20,001^3 (8 T cells)</td>
<td>12K</td>
</tr>
<tr>
<td>Dawn</td>
<td>BG/P</td>
<td>15,871^3 (4 T cells)</td>
<td>64K</td>
</tr>
<tr>
<td>Franklin</td>
<td>Cray XT4</td>
<td>12,596^3 (2 T cells)</td>
<td>32K</td>
</tr>
<tr>
<td>JaguarPF</td>
<td>Cray XT5</td>
<td>12,596^3 (2 T cells)</td>
<td>32K</td>
</tr>
<tr>
<td>Juno</td>
<td>X86_64</td>
<td>10,000^3 (1 T cells)</td>
<td>16K</td>
</tr>
<tr>
<td>Franklin</td>
<td>Cray XT4</td>
<td>10,000^3 (1 T cells)</td>
<td>16K</td>
</tr>
<tr>
<td>Ranger</td>
<td>Sun</td>
<td>10,000^3 (1 T cells)</td>
<td>16K</td>
</tr>
<tr>
<td>Purple</td>
<td>IBM P5</td>
<td>8,000^3 (0.5 T cells)</td>
<td>8K</td>
</tr>
</tbody>
</table>

*Scaling Studies of Isosurface Extraction and Volume Rendering (2009)*

VisIt is also used daily by domain scientists.
The VisIt team focuses on making a robust, usable product for end users.

- Regular releases (~ 6 / year)
  - Executables for all major platforms
  - End-to-end build process script `build_visit`

- User Support and Training
  - visitusers.org, wiki for users and developers
  - Email lists: visit-users, visit-developers
  - Beginner and advanced tutorials
  - VisIt class with detailed exercises

- Documentation
  - "Getting data into VisIt" manual
  - Python interface manual
  - Users reference manual

Slides from the VisIt class
VisIt provides a flexible data model, suitable for many application domains.

- **Mesh Types:**
  - Point, Curve, 2D/3D Rectilinear, Curvilinear, Unstructured
  - Domain Decomposed, AMR
  - Time Varying
  - Primarily linear element support, limited quadratic element support

- **Fields:**
  - Scalar, Vector, Tensor, Material volume fractions, Species

VisIt currently supports over 110 file formats.
VisIt employs a parallelized client-server architecture.

Local Components

- VisIt Viewer
  - VisIt GUI
  - VisIt CLI
  - Python Clients
  - Java Clients

Parallel Cluster

- VisIt Engine
  - Data Plugin

MPI

- VisIt Engine
  - Data Plugin

Data Flow Network

- Filter
- Filter
- Filter

Data

(Data or Files)

Project Introduction
VisIt automatically switches to a scalable rendering mode for large data sets.

- Rendering Modes:
  - Local (hardware)
  - Remote (software or hardware)

- Beyond surfaces:
  - VisIt also provides scalable volume rendering.
VisIt’s infrastructure provides a flexible platform for custom workflows.

- **C++ Plugin Architecture**
  - Custom File formats, Plots, Operators
  - Interface for custom GUIs in Python, C++ and Java

- **Python Interfaces**
  - Python scripting and batch processing
  - Data analysis via Python Expressions and Queries.

- **Libsim library**
  - Enables coupling of simulation codes to VisIt for in situ visualization.
VisIt is used as a platform to deploy visualization research.

- **Research Collaborations:**
  - 2006 – 2011
  - 2012 – 2017

- **Research Focus:**
  - Next Generation Architectures
  - Parallel Algorithms
  - In-Situ Processing

- **Algorithms research:**
  - How to efficiently calculate particle paths in parallel.

- **Scaling research:**
  - Scaling to 10Ks of cores and trillions of cells.

- **Methods research:**
  - How to incorporate statistics into visualization.

- **Reconstructed material interfaces for visualization**

- **Visualization and Analysis Enabled Materialization of Multilevel Data**
VisIt: What’s the Big Deal?

- Everything works at scale
- Robust, usable tool
- Features that span the “power of visualization”:
  - Data Exploration
  - Confirmation
  - Communication
- Features for different kinds of users:
  - Visualization Experts
  - Code Developers
  - Code Consumers

Healthy future: Vibrant Developer and User Communities
Resources

- **User resources:**
  - Main website: [http://www.llnl.gov/visit](http://www.llnl.gov/visit)
  - Wiki: [http://www.visitusers.org](http://www.visitusers.org)
  - Email: [visitusers@ornl.gov](mailto:visitusers@ornl.gov)

- **Development resources:**
  - Email: [visit-developers@ornl.gov](mailto:visit-developers@ornl.gov)
  - SVN: [http://portal.nersc.gov/svn/visit](http://portal.nersc.gov/svn/visit)
Hands On Visualizations
30 minute Hands on visualization of a Blood Flow Simulation.

In-depth hands on visualization of a Water Flow Simulation.

Additional Material for Lunch or Dinner Hands-on sessions:

[Supporting Slides]
Visualization Techniques for Mesh-based Simulations
Terminology

- **Meshes**: discretization of physical space
  - Contains “zones” / “cells” / “elements”
  - Contains “nodes” / “points” / “vertices”
    - VisIt speak: zone & node

- **Fields**: variables stored on a mesh
  - **Scalar**: 1 value per zone/node
    - Example: pressure, density, temperature
  - **Vector**: 3 values per zone/node (direction)
    - Example: velocity
      - Note: 2 values for 2D, 3 values for 3D
  - More fields discussed later…
Pseudocolor

- Maps scalar fields (e.g., density, pressure, temperature) to colors.
Contour / Isosurface
Volume rendering

VisIt can combine volume rendering and opaque geometry
Particle advection: the foundation of flow visualization

- Displace massless particle based on velocity field

- $S(t) = \text{position of curve at time } t$
  - $S(t_0) = p_0$
    - $t_0$: initial time
    - $p_0$: initial position
  - $S'(t) = v(t, S(t))$
    - $v(t, p)$: velocity at time $t$ and position $p$
    - $S'(t)$: derivative of the integral curve at time $t$

This is an ordinary differential equation
Streamlines

- Streamlines – instantaneous paths
- Pathlines – time dependent paths
Meshes

- All data in VisIt lives on a mesh
- Discretizes space into points and cells
  - (1D, 2D, 3D) + time
  - Topological dimension need not match spatial dimension (e.g. 2D surface in 3D space)
- Provides a place for data to be located
- Defines how data is interpolated
Variables

- Scalars, Vectors, Tensors
- Associated with points or cells of a mesh
  - Points: linear interpolation
  - Cells: piecewise constant
- Can have different dimensionality than the mesh (e.g. 3D vector data on a 2D mesh)
Materials

- Describes disjoint spatial regions at a sub-grid level
- Volume/area fractions
- VisIt will do high-quality sub-grid material interface reconstruction
Species

- Similar to materials, describes sub-grid variable composition
  - Example: *Material “Air” is made of species “N₂”, “O₂”, “Ar”, “CO₂”, etc.*

- Used for mass fractions

- Generally used to weight other scalars (e.g. partial pressure)
Parallel Meshes

- Provides aggregation for meshes
- A mesh may be composed of large numbers of mesh “blocks”
- Allows data parallelism
AMR meshes

- Mesh blocks can be associated with patches and levels
- Allows for aggregation of meshes into AMR hierarchy levels
AMR Example: Image vs. Data Resolution
Data representation for mesh-based HPC simulations
VisIt’s Core Abstractions
VisIt’s core abstractions

- **Databases**: How datasets are read
- **Plots**: How you render data
- **Operators**: How you manipulate data
- **Expressions**: Mechanism for generating derived quantities
- **Queries**: How to access quantitative information
Examples of VisIt Pipelines

- Databases: how you read data
- Plots: how you render data
- Operators: how you transform/manipulate data
- Expressions: how you create new fields
- Queries: how you pull out quantitative information

Database

Open a database, which reads from a file
(example: open file1.hdf5)

Plot

Make a plot of a variable in the database
(example: Volume plot)
Examples of VisIt Pipelines

- **Databases**: how you read data

- **Plots**: how you render data

- **Operators**: how you transform/manipulate data

- **Expressions**: how you create new fields

- **Queries**: how you pull out quantitative information

- **Database**: Open a database, which reads from a file (example: open file1.hdf5)

- **Operator**: Apply an operator to transform the data (example: Slice operator)

- **Plot**: Plot a variable in the database (example: Pseudocolor plot)
Examples of VisIt Pipelines

- Databases: how you read data
- Plots: how you render data
- Operators: how you transform/manipulate data
- Expressions: how you create new fields
- Queries: how you pull out quantitative information

Open a database, which reads from a file (example: open file1.hdf5)

Apply an operator to transform the data (example: Slice operator)

Apply a second operator to transform the data (example: Elevate operator)

Plot a variable in the database (example: Pseudocolor plot)
Examples of VisIt Pipelines

- **Databases:** how you read data
  - Open a database, which reads from a file (example: open file1.hdf5)

- **Plots:** how you render data
  - Create derived quantities from fields in the file (example: magnitude(velocity))
  - Plot the expression variable (example: Pseudocolor plot)

- **Operators:** how you transform/manipulate data

- **Expressions:** how you create new fields

- **Queries:** how you pull out quantitative information
Examples of VisIt Pipelines

- Databases: how you read data
- Plots: how you render data
- Operators: how you transform/manipulate data
- Expressions: how you create new fields
- Queries: how you pull out quantitative information

**Database**
- Open a database, which reads from a file (example: open file1.hdf5)

**Plot**
- Plot a field from the file (example: density + Pseudocolor plot)

**Query**
- Extract quantitative information (example: integrate density to find mass)
Examples of VisIt Pipelines

- Databases: how you read data
  - Open a database, which reads from a file (example: open file1.hdf5)

- Plots: how you render data
  - Create derived quantities from fields in the file (example: magnitude(velocity))

- Operators: how you transform/manipulate data
  - Apply an operator to transform the data (example: Slice operator)

- Expressions: how you create new fields
  - Apply a second operator to transform the data (example: Elevate operator)

- Queries: how you pull out quantitative information
  - Plot a field (example: speed + Pseudocolor plot)
  - Extract quantitative information (example: maximum speed over cross-section)
Practical Tips for Using VisIt
Practical Tips for Using VisIt

- How to get VisIt to read your data
- How to get help when you run into trouble
How to get VisIt to read your data.

- There is an extensive manual on this topic: “Getting Data Into VisIt”
  
  https://wci.llnl.gov/simulation/computer-codes/visit/manuals

- Three ways:
  - Use a known format
  - Write a file format reader
  - In situ processing
File formats that VisIt supports

- **110+ Total Readers**: ADIOS, BOV, Boxlib, CCM, CGNS, Chombo, CLAW, EnSight, ENZO, Exodus, FLASH, Fluent, GDAL, Gadget, Images (TIFF, PNG, etc), ITAPS/MOAB, LAMMPS, NASTRAN, NETCDF, Nek5000, OpenFOAM, PLOT3D, PlainText, Pixie, Shapefile, Silo, Tecplot, VTK, Xdmf, Vs, and many more


- Some readers are more robust than others.
  - For some formats, support is limited to flavors of a file a VisIt developer has encountered previously (e.g. Tecplot).
File formats that VisIt supports

- 110+ Total Readers: ADIOS, BOV, Boxlib, CCM, CGNS, Chombo, Exodus, FLASH (TIFF, PNG, etc), NASTRAN, NETCDF, PLOT3D, Plaintext, Tecplot, VTK, and many more.


- Some readers are more robust than others.
  - For some formats, support is limited to flavors of a file a VisIt developer has encountered previously (e.g. Tecplot).
Application Code Formats

- ANSYS
- Cale
- CASTRO
- CCM
- DDCMD
- Dyna3D
- Enzo
- FLASH
- FVCOM
- Gadget
- LAMMPS
- NASTRAN
- Nek5000
- OVERFLOW
- PATTRAN
- Pixie
- S3D
- ZeusMP
Application Toolkit Formats

- Adventure I/O
- BoxLib
- Chombo
- ITAPS
- OpenFOAM
- SAMRAI
- Spheral
General Scientific Data Formats

- ADIOS
- CGNS
- Exodus
- HDF5
- H5Part
- NETCDF
- PDB
- Silo
- XDMF

Common Structure

Conventions

API

Array Storage I/O

Silo / Ale3d
File formats that VisIt supports

- Common array writing libraries:
  - NETCDF
    - VisIt reader understands many (but not all) conventions
  - HDF5
    - Pixie is most general HDF5 reader
    - Many other HDF5 readers

- Xdmf: specify an XML file that describes semantics of arrays in HDF5 file

- VizSchema (Vs): add attributes to your HDF5 file that describes semantics of the arrays.
Silo file format

- Silo is a mature, self-describing file format that deals with multi-block data.
- It has drivers on top of HDF5 and “PDB”.
- Fairly rich data model
- More information:
  - [https://wci.llnl.gov/simulation/computer-codes/silo](https://wci.llnl.gov/simulation/computer-codes/silo)
Welcome to Silo

A mesh and field I/O library and scientific database

- Structured Rectilinear Mesh
- Gridless Point Mesh
- Structured (Curvilinear) Mesh
- Arbitrary Subsets
- Silex browser for Silo files
- Constructive Solid Geometry (CSG) Mesh
- Unstructured Zoo (UCD) Mesh
- Adaptive Mesh Refinement (AMR) Mesh
- Mixing Materials
- Arbitrary Polyhedral Mesh
- XY Curve
Specialized Scientific Data Formats

- BOW
- FITS
- GDAL
- MatrixMarket
- ProteinDataBank
- ESRI Shapefile
- XYZ

DEM from GDAL

Protein Data Bank
Visualization Formats

- VTK
- EnSight
- GMV
- Plot3D
- Tecplot
- Vis5D
- Xmdv
VTK File Format

- The VTK file format has both ASCII and binary variants.

- Easiest way to write VTK files: use VTK modules
  - ... but this creates a dependence on the VTK library

- You can also try to write them yourself, but this is an error prone process.

- Third option: visit_writer
VisIt Writer writes VTK files

- It is a “library” (actually a single C file) that writes VTK-compliant files.
  - The typical path is to link visit_writer into your code and write VTK files

- There is also a Python binding for visit_writer.
  - The typical path is to write a Python program that converts from your format to VTK

- Both options are short term: they allow you to play with VisIt on your data. If you like VisIt, then you typically formulate a long term file format strategy.

- More information on visit_writer:
import visit_writer
import math
import sys

nX = 20
nY = 20
conn = []
for i in range(nX-1):
    for j in range(nY-1):
        pt1 = j*(nX) + i;
        pt2 = j*(nX) + i+1;
        pt3 = (j+1)*(nX) + i+1;
        pt4 = (j+1)*(nX) + i;
        conn.append([ "quad", pt1, pt2, pt3, pt4 ])

pts = []
rad = []
for i in range(nX):
    for j in range(nY):
        pts.extend([ [float(i), float(j), 0 ]])
        rad.append( math.sqrt(i*i + j*j) )

var_datum = [ "radius", 1, 1, rad ]
vars = [ var_datum ]
visit_writer.WriteUnstructuredMesh("ugrid.vtk", 0, pts, conn, vars)
sys.exit()
Graphics Formats

- Image
  - (PNG, JPEG, TIFF, BMP, etc.)
- RAW
- STL
- Wavefront OBJ
General ASCII Data Formats

- Curve2D
- Lines
- PlainText
- Point3D
Practical Tips for Using VisIt

- How to get VisIt to read your data
- How to get help when you run into trouble
How to get help when you run into trouble

- **FAQ**
  - [https://wci.llnl.gov/simulation/computer-codes/visit/faq](https://wci.llnl.gov/simulation/computer-codes/visit/faq)

- **VisIt Users Mailing List**
  - Address: [visit-users@elist.ornl.gov](mailto:visit-users@elist.ornl.gov)
  - Info: [https://elist.ornl.gov/mailman/listinfo/visit-users](https://elist.ornl.gov/mailman/listinfo/visit-users)
  - Archive: [https://elist.ornl.gov/pipermail/visit-users/](https://elist.ornl.gov/pipermail/visit-users/)

- **VisIt Users Wiki**
  - [http://www.visitusers.org](http://www.visitusers.org)

- **VisIt Users Forum**
  - [http://visitusers.org/forum/YaBB.pl](http://visitusers.org/forum/YaBB.pl)

- **Priority support for specific user groups:**
  - VisIt-help-{XYZ} Mailing Lists

- **Reference Manuals**
  - [https://wci.llnl.gov/simulation/computer-codes/visit/manuals](https://wci.llnl.gov/simulation/computer-codes/visit/manuals)
FAQ: https://wci.llnl.gov/simulation/computer-codes/visit/faqs
VisIt-users Mailing List

- You may only post to mailing list if you are also a subscriber.
- Approximately 400 recipients, approx. 300 posts per month.
- Developers monitor mailing list, strive for 100% response rate.
- Response time is typically excellent (O(1 hour)).
  - International community participates … not unusual for a question from Australia to be answered by a European, while all US developers are asleep.
- List Address: visit-users@ornl.gov
- More information: https://email.ornl.gov/mailman/listinfo/visit-users
- Archive: https://email.ornl.gov/pipermail/visit-users/
VisItusers.org

- Great source for VisIt tips and recipes.
- Users section has lots of practical advice:
  - “I solved this problem using this technique”
  - “Here’s my script to do this analysis”

VisItusers.org is the VisIt project’s staging area for usage recipes and future formal documentation.
VisIt Users Forum

- [http://www.visitusers.org/forum](http://www.visitusers.org/forum)

- Increasingly popular option; you can post without receiving 300 emails a month
  - But it is viewed by less people and less well supported.

- Google indexes these pages.
Visit-help-{XYZ}

- Some customer groups pay for priority VisIt support:
  - These customers can post directly to specific visit-help-{XYZ} support lists without subscribing.
  - The messages are received by all VisIt developers and supported collectively.

- Example Lists:
  - visit-help-asc, visit-help-scidac
Manuels & Other Documentation

- Getting Started Manual
- Users Manual
- Python Interface
- Getting Data Into VisIt
- VisIt Class Slides
- VisIt Class Exercises
- {Tutorials}
Resources

- **Presenters:**
  - Cyrus Harrison: cyrush@llnl.gov

- **User resources:**
  - Main website: [http://www.llnl.gov/visit](http://www.llnl.gov/visit)
  - Wiki: [http://www.visitusers.org](http://www.visitusers.org)
  - Email: visitusers@ornl.gov

- **Development resources:**
  - Email: visit-developers@ornl.gov
  - SVN: [http://portal.nersc.gov/svn/visit](http://portal.nersc.gov/svn/visit)