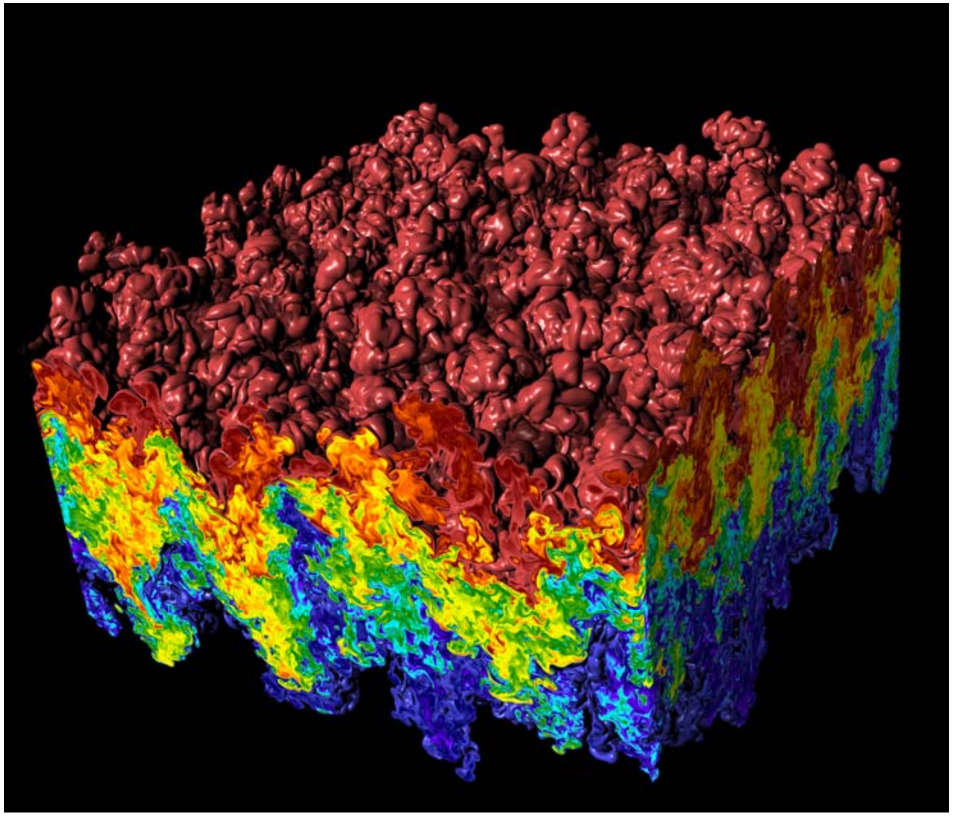
VisIt Overview

Hank Childs

VACET: Chief SW Engineer
ASC: V&V Shape Char. Lead

Supercomputing 2006
Tampa, Florida
November 13, 2006



27B element
Rayleigh-Taylor Instability
(MIRANDA, BG/L)

This is
UCRL-PRES-226373

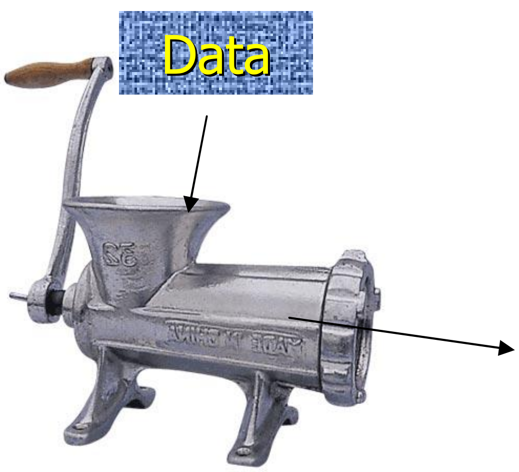
Work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract W-7405-Eng-48.
Lawrence Livermore National Laboratory, P.O. Box 808, Livermore, Ca, 94551

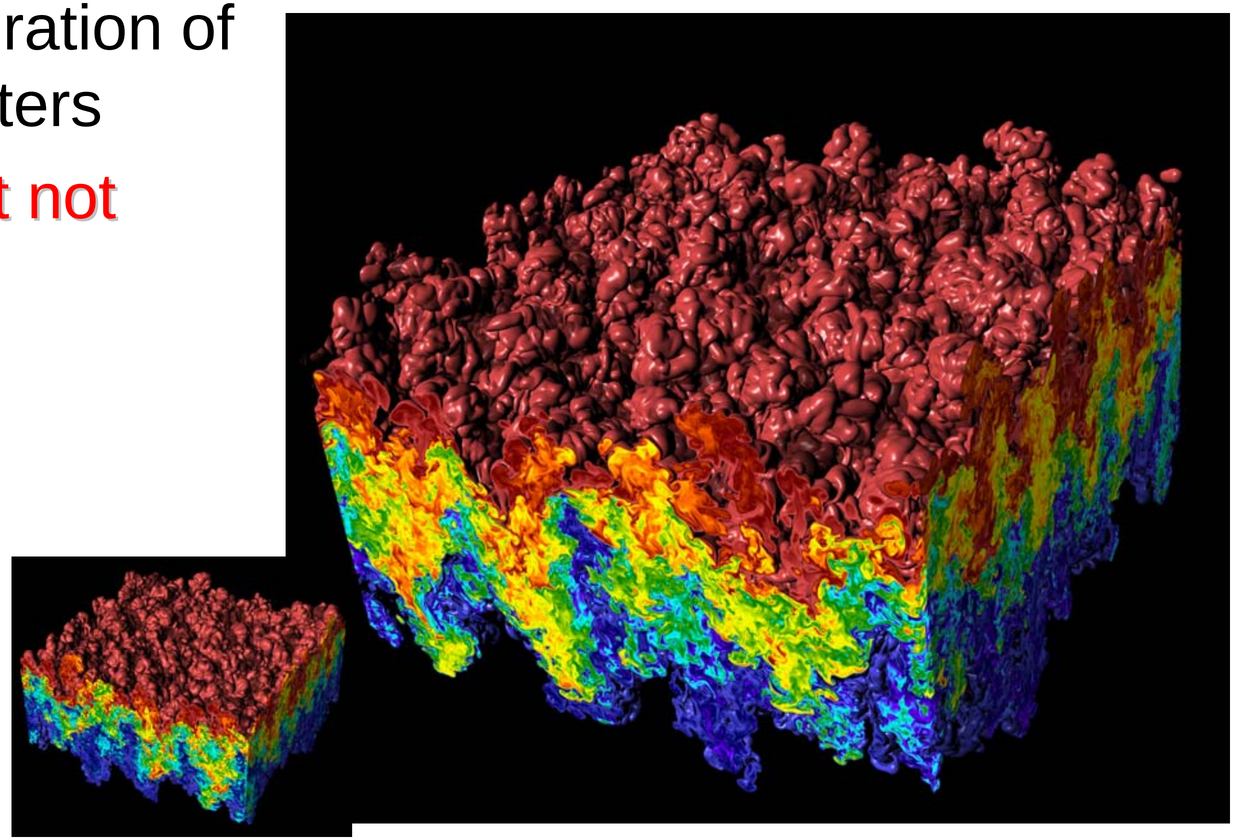
Petascale visualization: two incredible challenges!!

1) Address the incredible scale of data coming off the current generation of supercomputers

2) Provide features that improve the legibility of the data set

Necessary, but not sufficient!!





Outline

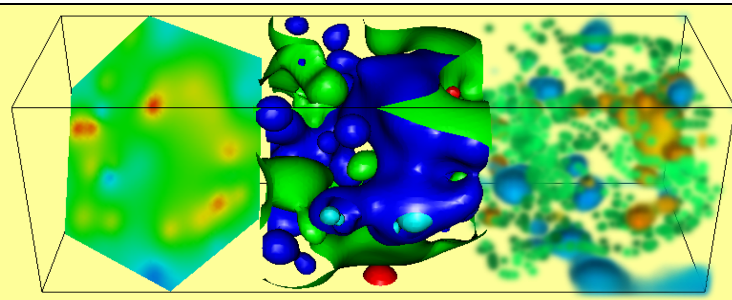
- Project Overview (brief)
- Architecture
- Challenge 1: Handling petascale data
- Challenge 2: Understanding petascale data

Outline

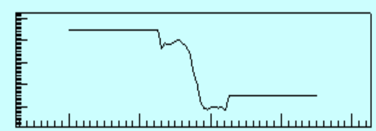
- Project Overview (brief)
- Architecture
- Challenge 1: Handling petascale data
- Challenge 2: Understanding petascale data

VisIt is a richly featured, turnkey application

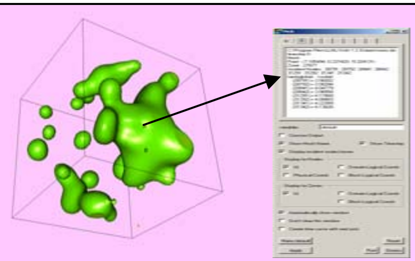
- VisIt is an open source, end user visualization and analysis tool for simulated and experimental data
  - Used by: physicists, engineers, code developers, vis experts
  - >20 simulation codes & 300 users at LLNL, >25K downloads on web
- Customer Requirements:



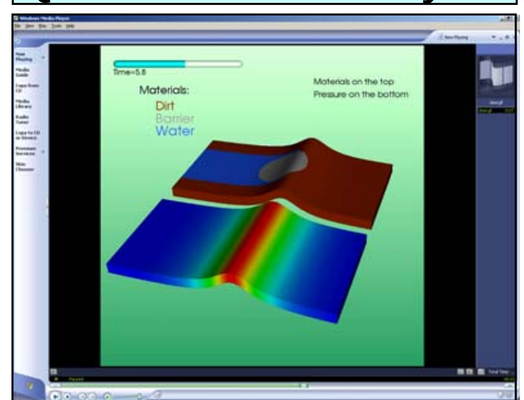
Data Exploration



Quantitative Analysis



Code Assessment



Presentations

VisIt Features (Highlights)

- Meshes: rectilinear, curvilinear, unstructured, point, AMR
- Data: scalar, vector, tensor, material, species
- Dimension: 1D, 2D, 3D, time varying
- Rendering: pseudocolor, volume rendering, hedgehogs, glyphs, mesh lines, etc…
- Data Manipulation: slicing, contouring, thresholding, clipping, restrict to box, reflect, project, revolve, …
- Formats: over fifty readers
- Derived quantities: >100 interoperable building blocks

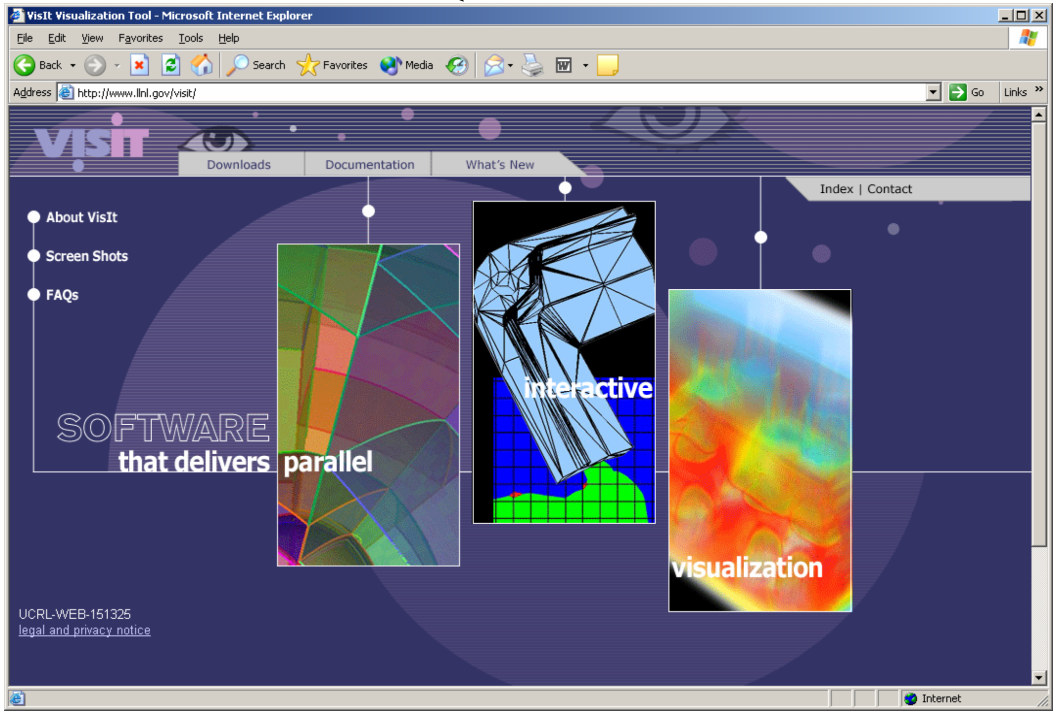
  +,-,*,/, gradient, mesh quality, if-then-else, and, or, not
- Many general features: position lights, make movie, etc
- Data analysis: later in this talk

Who's contributing to VisIt?

- ASC: focus on data analysis
- SciDAC: focus on new application areas, AMR
- Additional organizations (last three months):
  - ORNL: molecular visualization
    performance enhancements
    improved file open capabilities
    build improvements
  - LANL: stereo rendering for parallel rendering
    parallel hardware accelerated volume rendering
  - AWE: bug fixes
  - Cray: static library support
  - MIT: NetCDF reader
  - NETL: Fluent reader

License, Availability

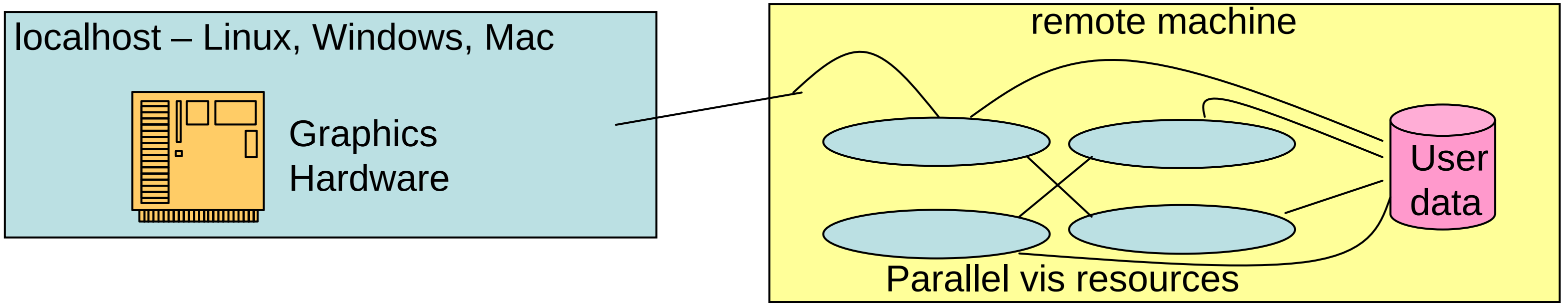
- Publicly available (http://www.llnl.gov/visit).
  - Website likely to change soon
- Open source/BSD
- World accessible repository coming soon
- Support for Linux, AIX, Tru64, Solaris,
  IRIX, Windows, and Mac.
  Cray coming soon.



Outline

- Project Overview (brief)
- Architecture
- Challenge 1: Handling petascale data
- Challenge 2: Understanding petascale data

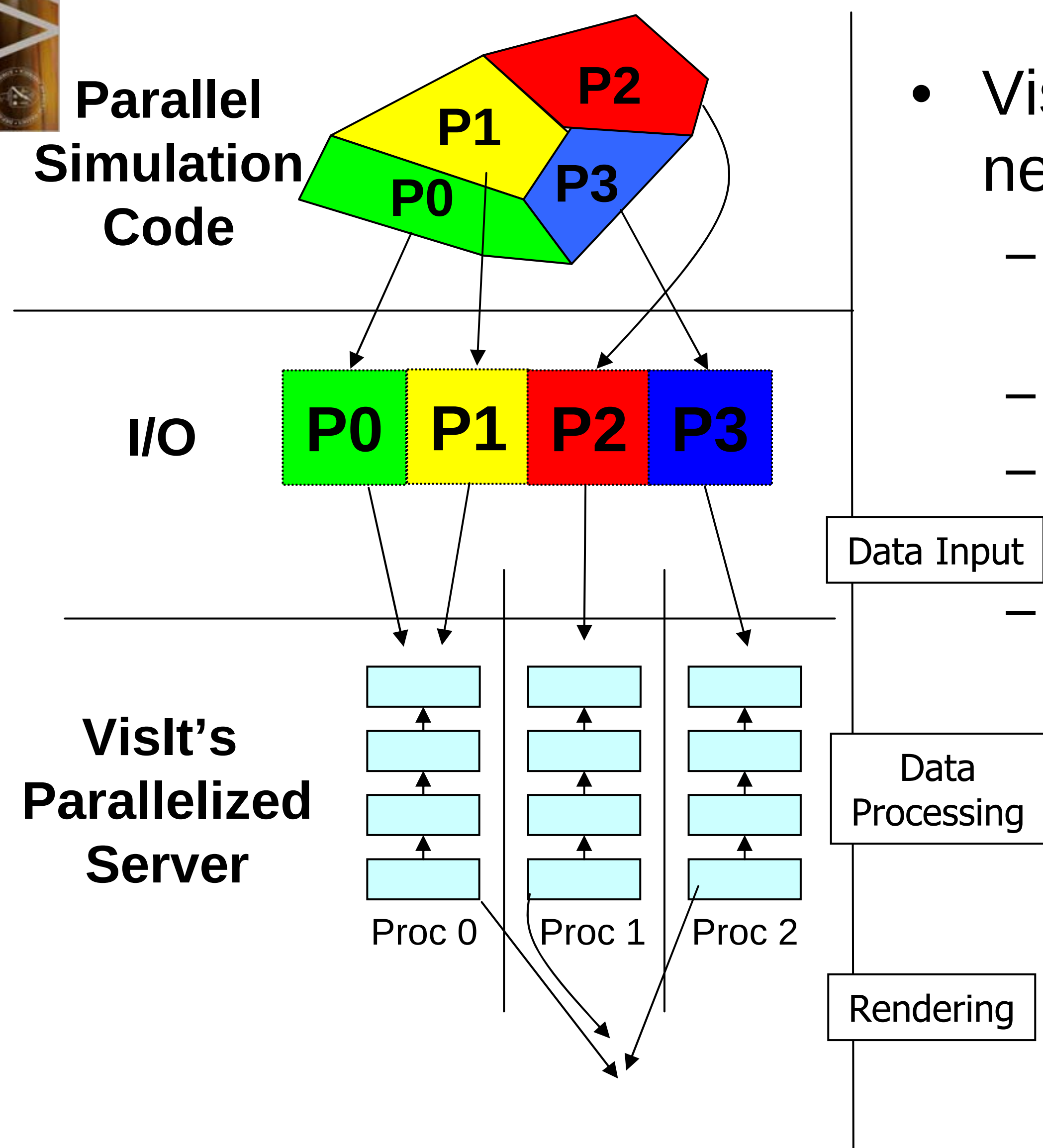
Architecture Summary



- Client-server observations:
  - Good for remote visualization
  - Leverages available resources
  - Scales well
  - No need to move data
- Additional design considerations:
  - Plugins
  - Multiple UIs: GUI (Qt), CLI (Python), more…
  - Third party libraries: VTK, Qt, Python, Mesa, +I/O libs

The principal architectural decisions – client/server, VTK, Qt, Python, plugins, etc – were put in place 6 years ago
…
VisIt has been building on that base ever since.

Parallelization covers data input, data processing, and rendering.



- VisIt: identical data flow networks on each processor.
  - Networks differentiated by portion of data they operate on.
  - Processing happens in parallel
  - Rendering: parallelized if necessary
  - "Scatter/gather"

Lots of ignored issues here: partitioning, communication, etc

Outline

- Project Overview (brief)
- Architecture
- Challenge 1: Handling petascale data
- Challenge 2: Understanding petascale data

The challenges of visualizing and analyzing peta-scale data

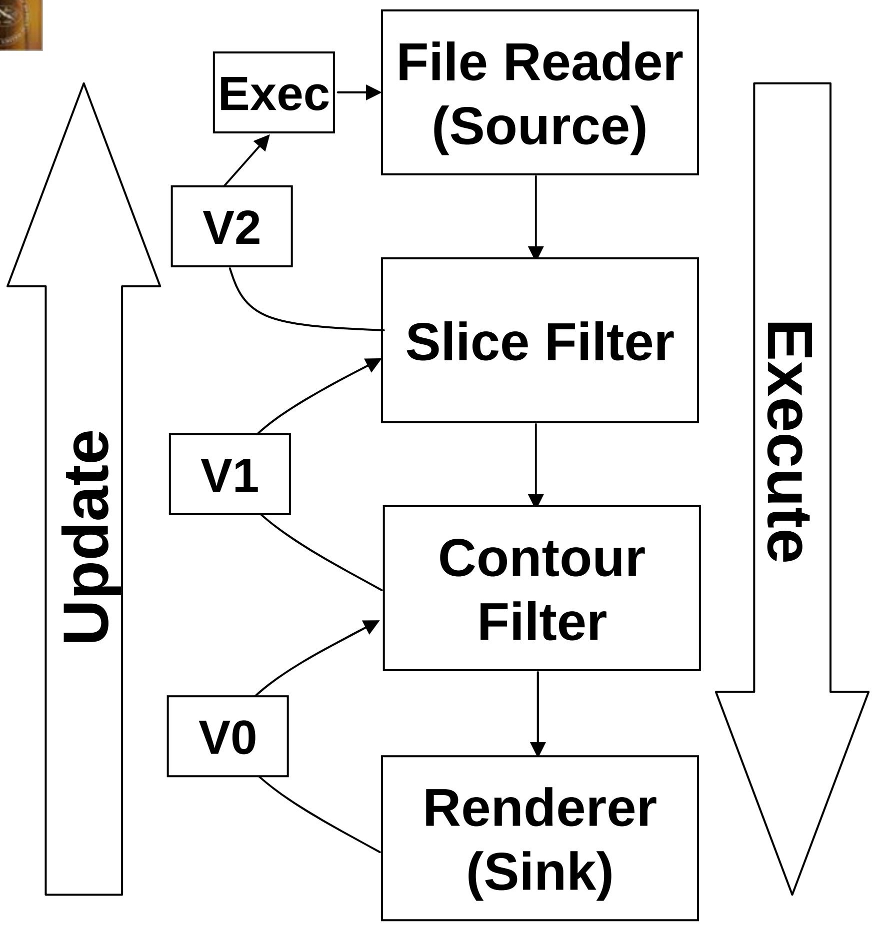
- Huge data→
  - Problems getting off disk
  - Problems storing in memory
  - Too slow to process the data?
  - Problems interpreting

- Solutions?
  - Parallelize
  - Analytics
  - In-line processing
  - Incorporate optimizations
    - Streaming
    - Only operate on relevant data
    - Multi-res

Issue lurking:
Not all optimizations play well with all features.
(This issue is exacerbated in richly featured tools.)

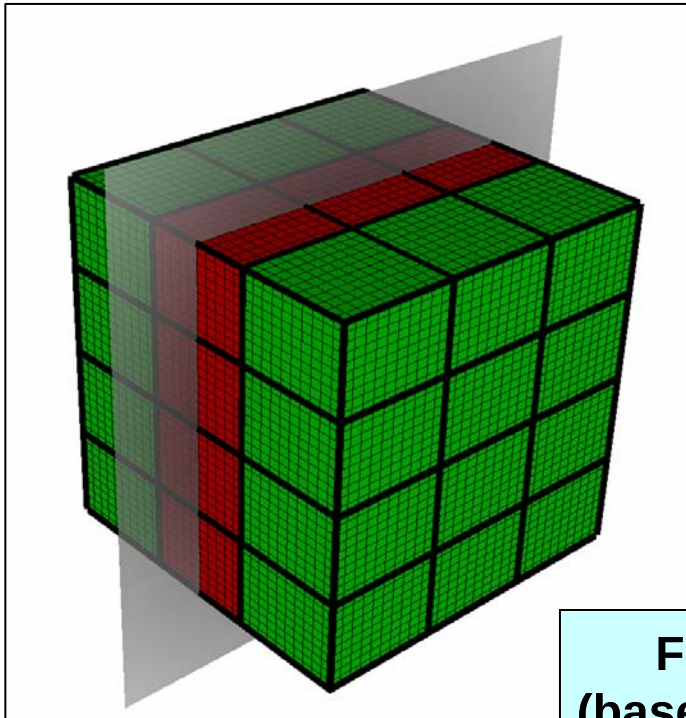
How do you manage which optimizations are appropriate at which time?

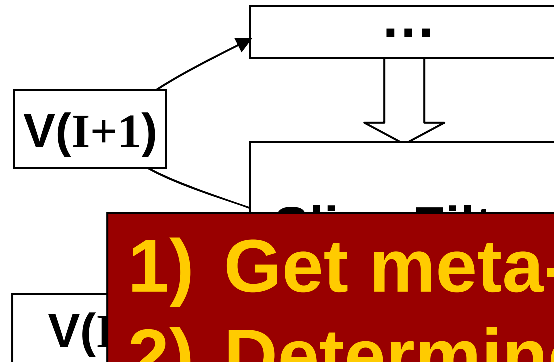
Contracts are the method for dynamically managing optimizations

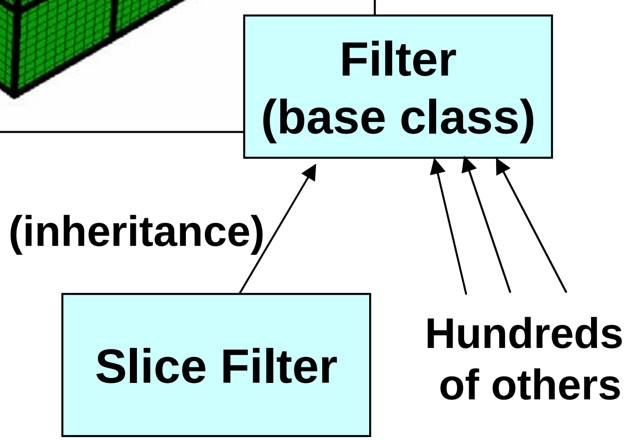


- Contracts are an extension to the basic data flow network design.
- Example Optimizations:
  - What data is operated on
  - How that data flows through pipeline
  - Ghost data creation
- See: Childs, et al, Vis05

Contracts allow you to work on only the relevant data.





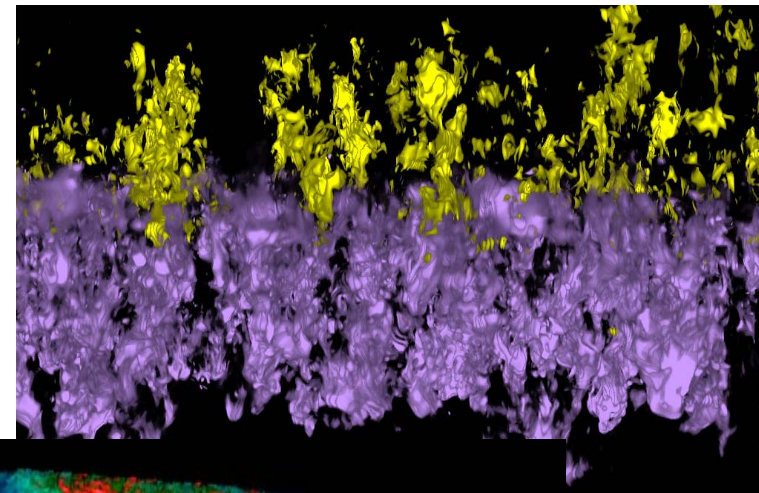


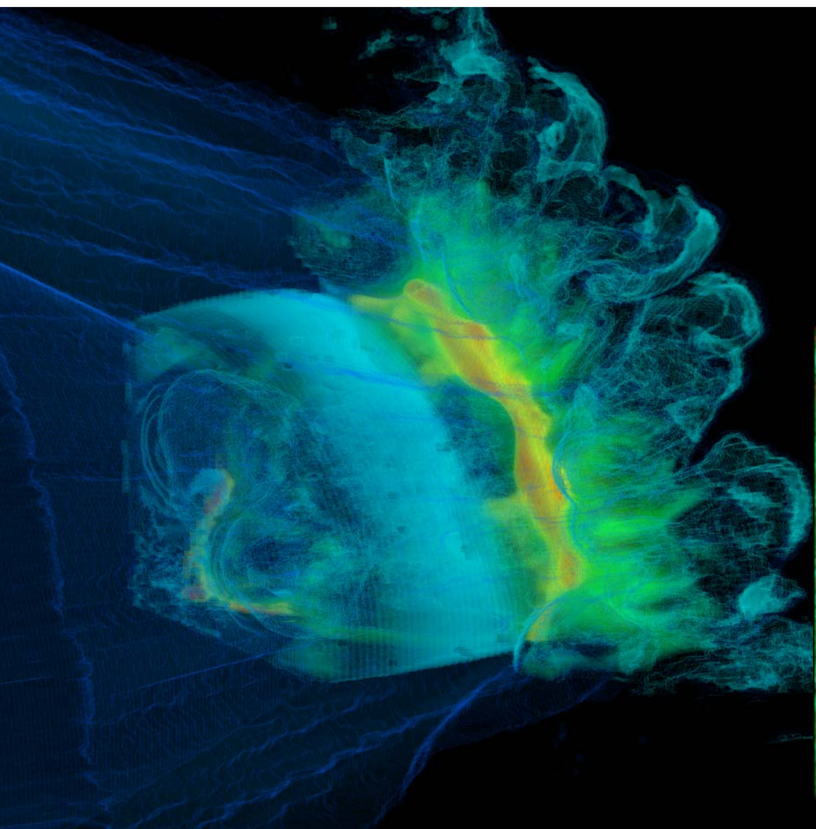
1) Get meta-data
2) Determine chunks that intersect slice
3) Restrict list of chunks to process in V(I+1)

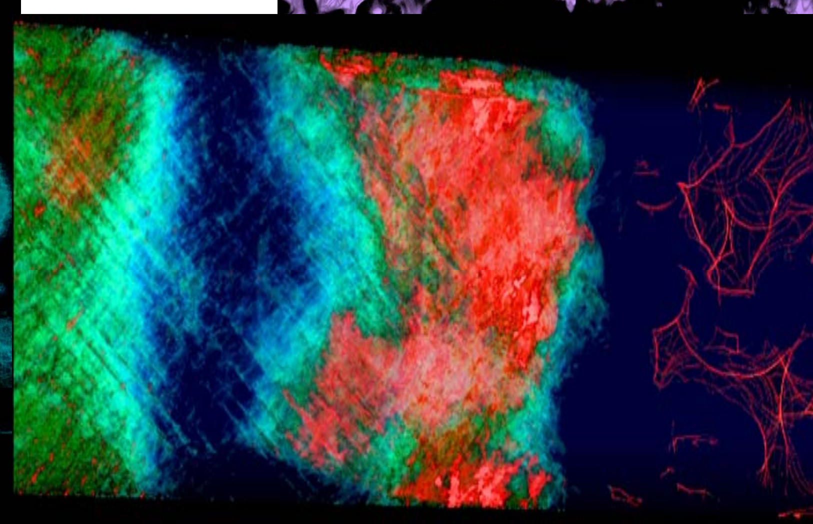
```
{
    return V(I+1);
}
```

Further complication: parallel-oblivious vs parallel-aware algorithms

- Parallel ray casting: involves complicated data layout schemes
- See: Childs, Duchaineau, Ma, EGPGV06







Outline

- Project Overview (brief)
- Architecture
- Challenge 1: Handling petascale data
- Challenge 2: Understanding petascale data

Understanding petascale data

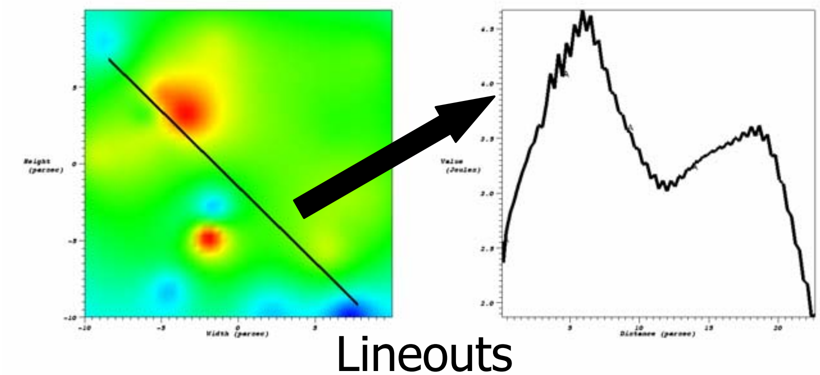
- New vis techniques are coming on line for big data
  - Two this afternoon:
    - Pascucci: topological characterization
    - Bethel: query driven visualization

- Plus data analysis

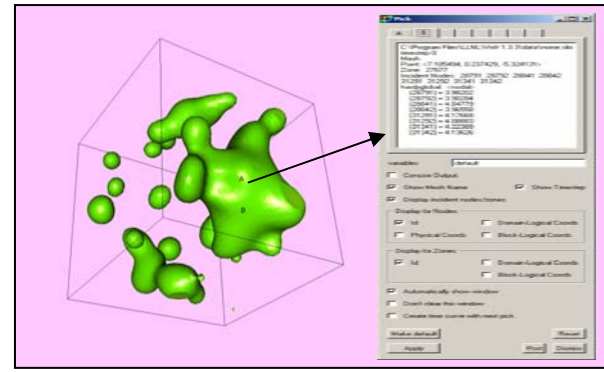
- Q: Why do both data analysis and vis in the same tool?
- A: They need the same assets
  - Scalable architecture for handling large data
  - Numerous data readers
  - Proper interpretation of data (ie material interface reconstruction)
  - Shared algorithms (multiple ways to slice/dice data)
  - <u>High flexibility and extensibility</u>

Data analysis casts a wide net.

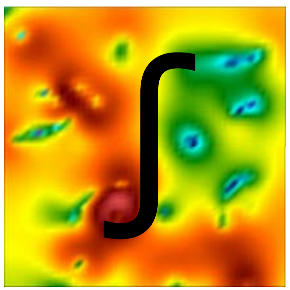
1) Techniques that span scientific domains



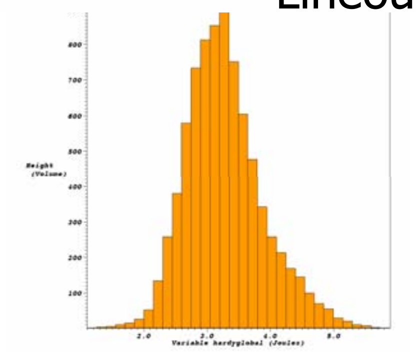
Lineouts



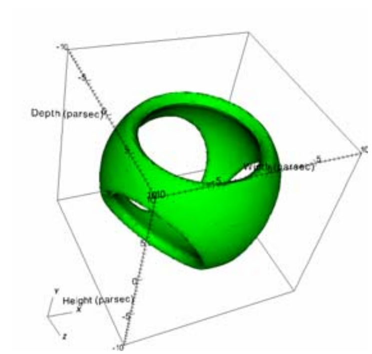
Picks



Integration



Histograms

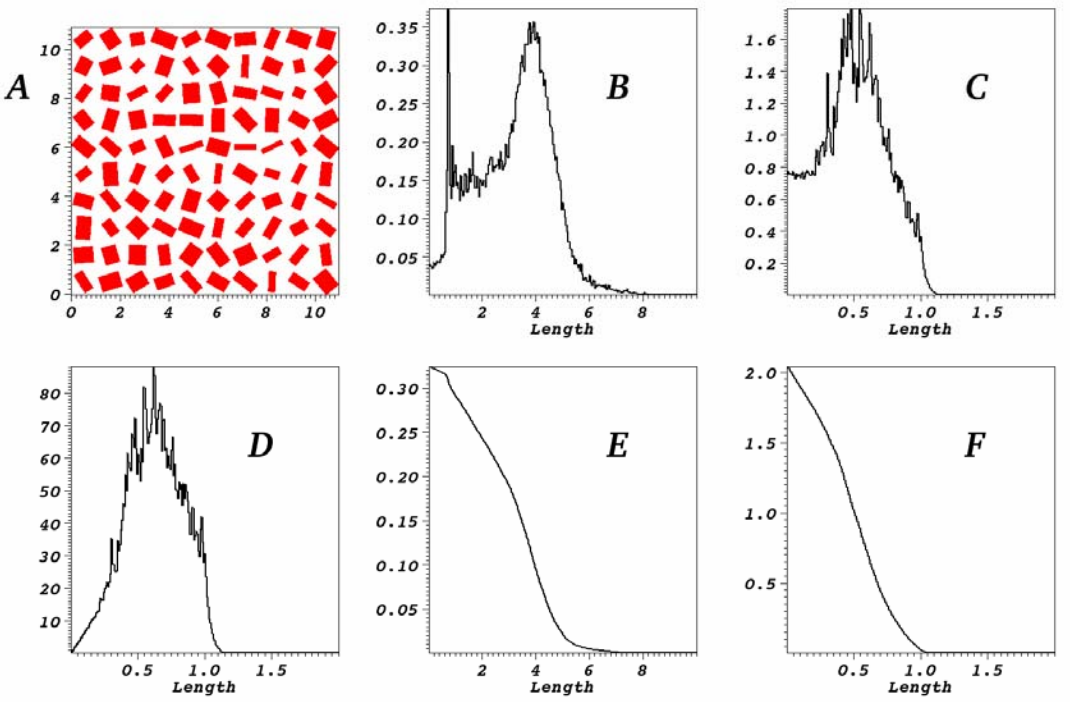


Volume
Surface Area
Centroid
Moments of Inertia

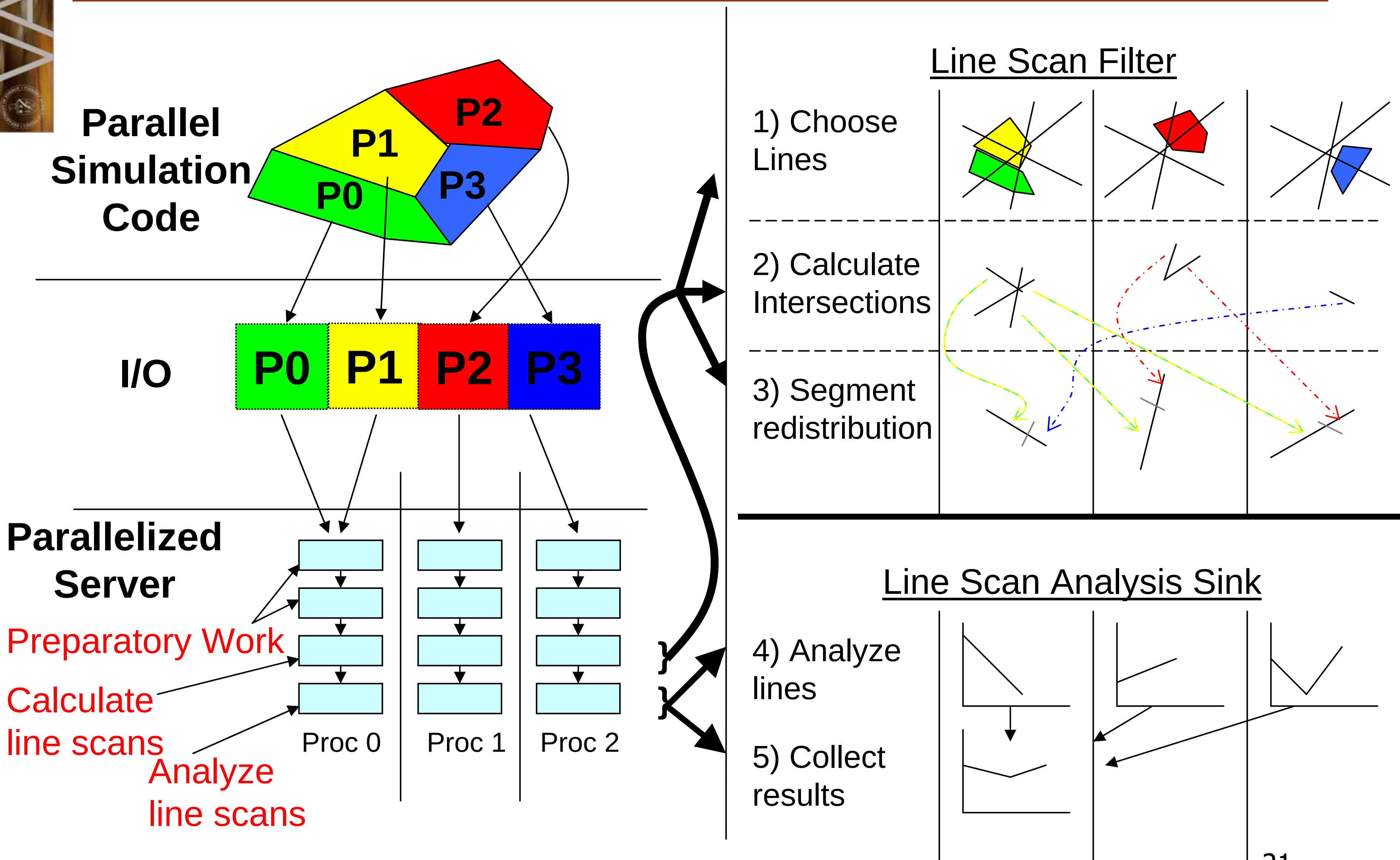
2) Techniques that are custom to scientific domains.
- Our architectures must be flexible enough to incorporate these.

Custom Analysis Example: Shape Characterization Metrics

- Strong interest in the characterization and comparison of shapes
- Simple metrics: volumes, how spherical, etc.
- Advanced metrics: topological characterization for comparative purposes (Pascucci talk later today)
- Advanced metrics (2):
  - chord length distribution
  - ray length distributions
  - mass as a function of length scale



EXAMPLE: chord length distributions place requirements on the underlying infrastructure



Outline

- Project Overview (brief)
- Architecture
- Challenge 1: Handling petascale data
- Challenge 2: Understanding petascale data
- Bonus: Comparative Vis

Petascale creates new challenges for comparative visualization.

- Comparisons are extremely important:
  - Compare simulation to experiment
  - Compare simulation to simulation
  - Compare one simulation over time
  - Compare symmetry conditions (one simulation, one time slice)

- Comparative visualization spans lots of techniques. Taxonomy:

  i. Image based ← view side by side

  ii. Data level ← next slide

  iii. Feature level ← Pascucci talk this afternoon

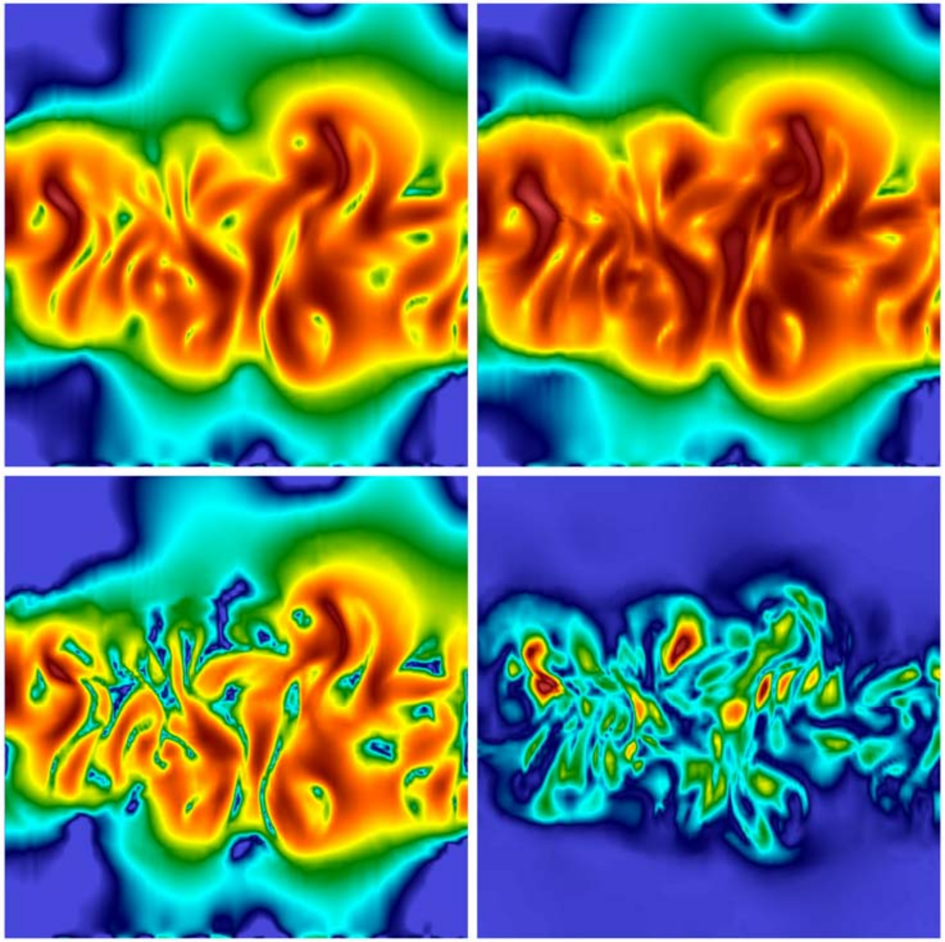
- Petascale uniqueness: parameter studies

Comparative techniques have applications with parameter studies/ensembles

Studying 25 Rayleigh-Taylor Instability calculations (all at 10us)
Two “knobs”: turbulent viscosity coefficient, buoyancy coefficient
Five values for each knob, 25 pairs total

Average Speed over all 25

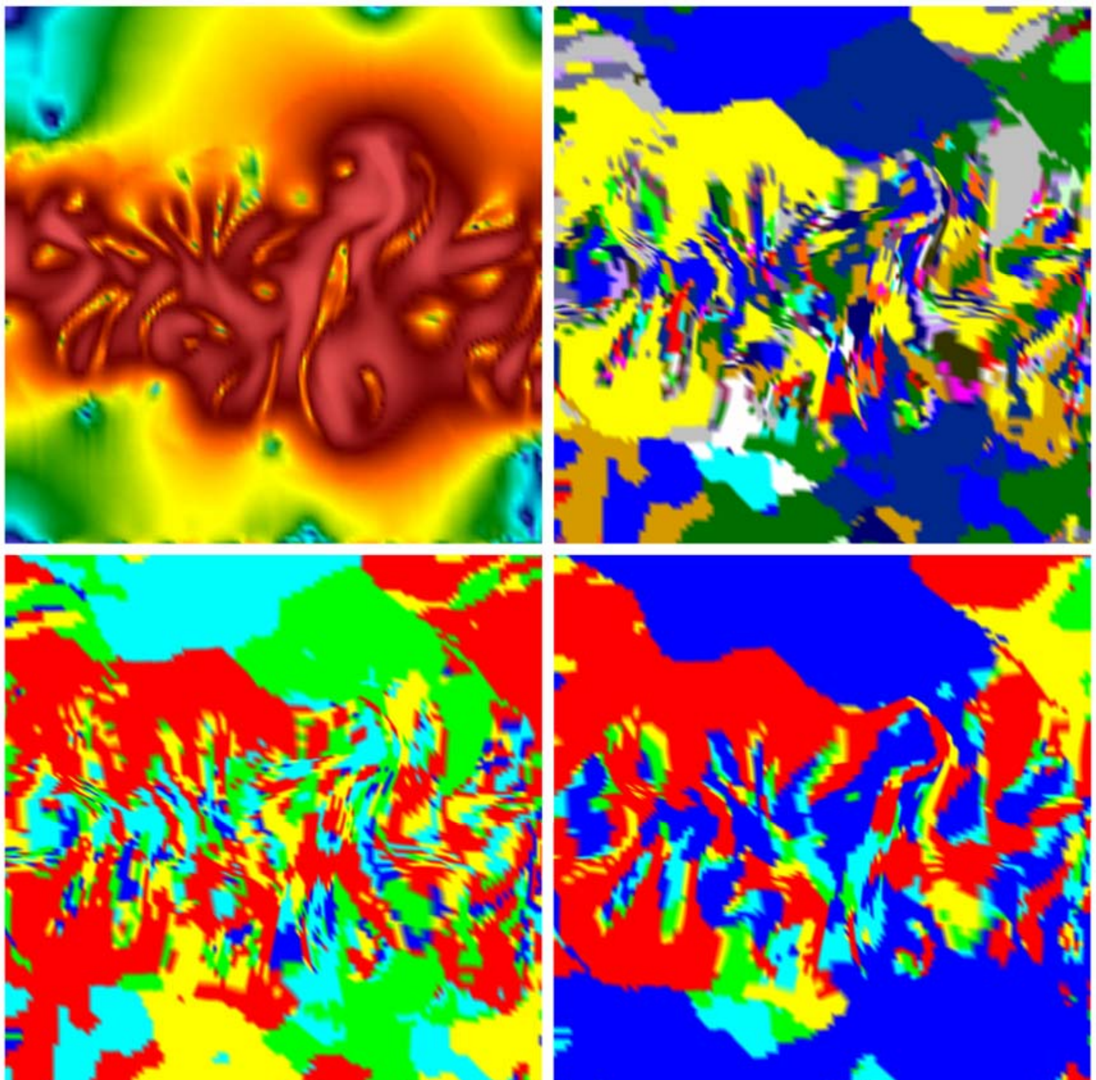
Max Speed over all 25



Min Speed over all 25

Biggest difference over all 25
(is this uncertainty quantification?)

Comparative techniques have applications with parameter studies/ensembles



Speed for one simulation

Coloring by Simulation ID with maximum speed

K0=V0, K1=V0 →

K0=V0, K1=V1 →

…

K0=V4, K1=V4 →

Coloring by "Knob 0" (buoyancy) with maximum speed

K0=V0→

K0=V1→

K0=V2→

K0=V3→

K0=V4→

Coloring by "Knob 1" (viscosity) with maximum speed

K1=V0→

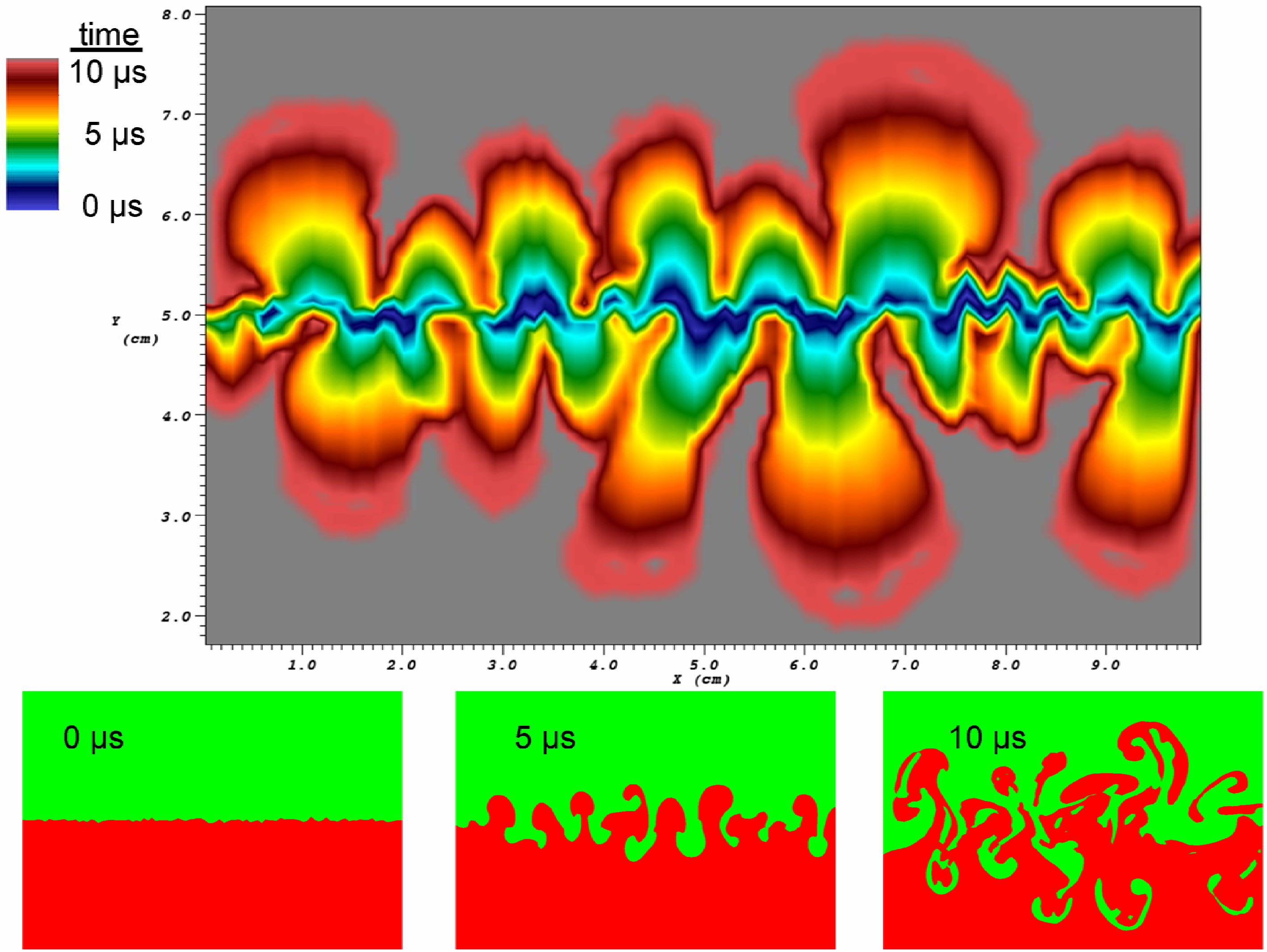
K1=V1→

K1=V2→

K1=V3→

K1=V4→

Data-level vis is deployed in derived quantity engine, allowing for very sophisticated analysis



Conclusion

- Two challenges:
  - Scale
  - Legibility
- VisIt is a relatively mature product that is well-positioned to leverage the community's upcoming results for petascale visualization

---

- Questions?
- Contact:
  - http://www.llnl.gov/visit
  - childs3@llnl.gov
  - visit-help@llnl.gov