Progressive Volume Rendering of Large Unstructured Grids

VIS 2006
BALTIMORE·MARYLAND·USA

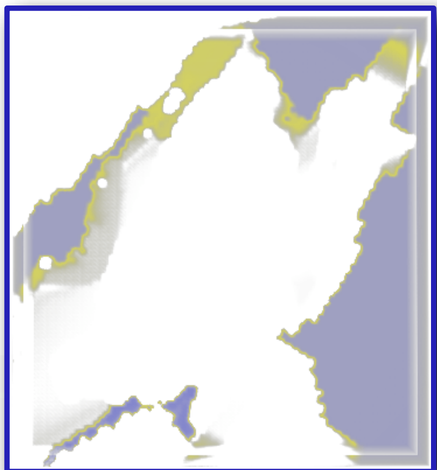
Steven P. Callahan[1], Louis Bavoil[1], Valerio Pascucci[2], and Cláudio T. Silva[1]

[1] SCI Institute, University of Utah
[2] Lawrence Livermore National Laboratory

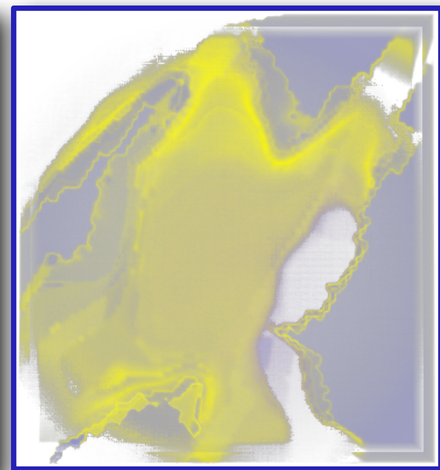
Motivation

- Large-scale simulations produce a lot of data
- Interactive visualization techniques not keeping up
- Meshes may be too large to render locally

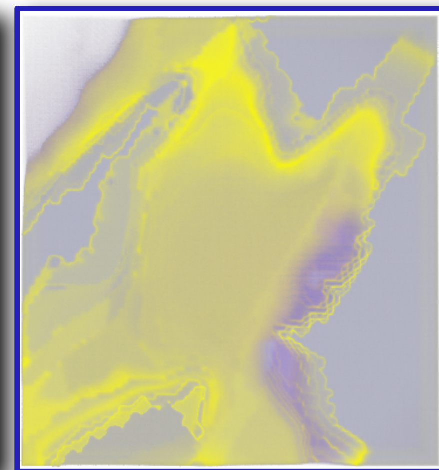
Progressive Volume Rendering



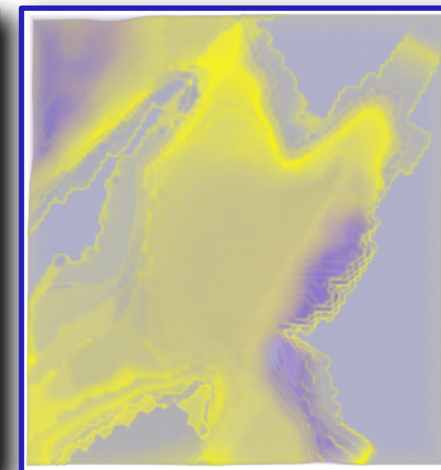
3%
0.01 sec



33%
7 sec



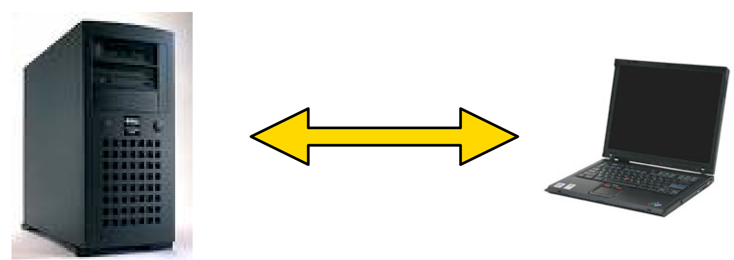
66%
18 sec



100%
34 sec

Objective

- Progressive Rendering
  - Show intermediate results
  - Reuse intermediate results
  - Allow user interrupt
  - Only render pertinent data
- Client-Server Architecture
  - Support a thin client with limited memory
  - Standard server used as a data repository
  - Facilitate remote visualization



Issues

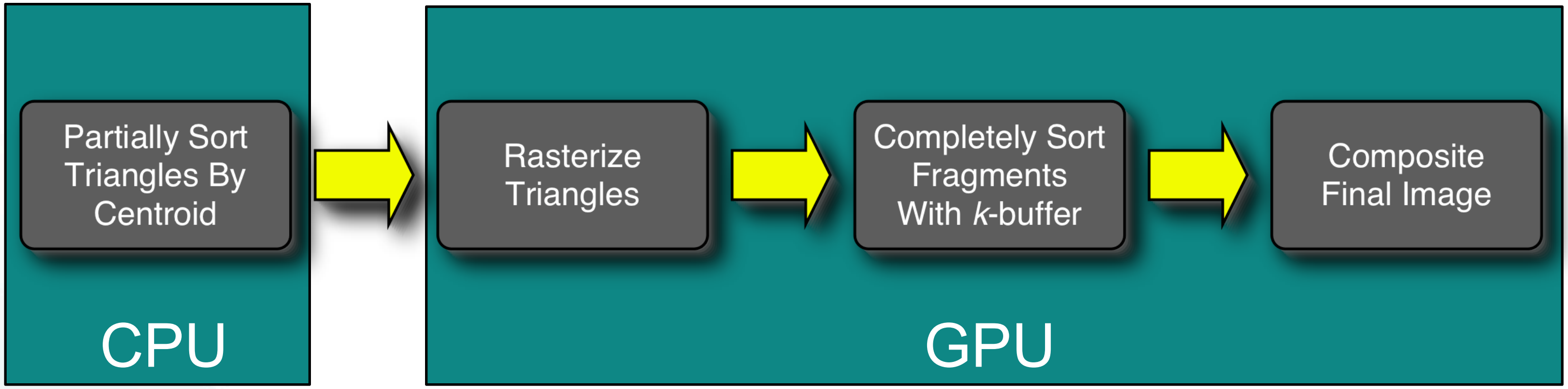
- Tetrahedra are not natively supported
  - Projected Tetrahedra
    - [Shirley and Tuchman ‘90, Wiley et al. ‘02]
- Compositing requires strict order
  - Visibility Sorting
    - [Williams et al. ‘92]
  - Ray Casting
    - [Bunyk et al. ‘97, Weiler et al. ‘03]
  - Hybrids
    - [Farias et al. ‘00, Callahan et al. ‘05]

Issues

- Hierarchical level-of-detail not suitable
  - Regular Sampling
    - [Leven et al. 2002]
  - Geometry Simplification
    - [Cignoni et al. 2005]
  - LOD Without Hierarchies
    - [Callahan et al. 2005]
- Remote Visualization difficult using a standard server
  - Image Transmission
    - [Engel et al. 2000]
  - Uncomposited Image Transmission
    - [Bethel et al. 2000]
  - Data Transmission
    - [Lippert et al. 1997, Engel et al. 1998, Kaehler et al. 2004]

Background

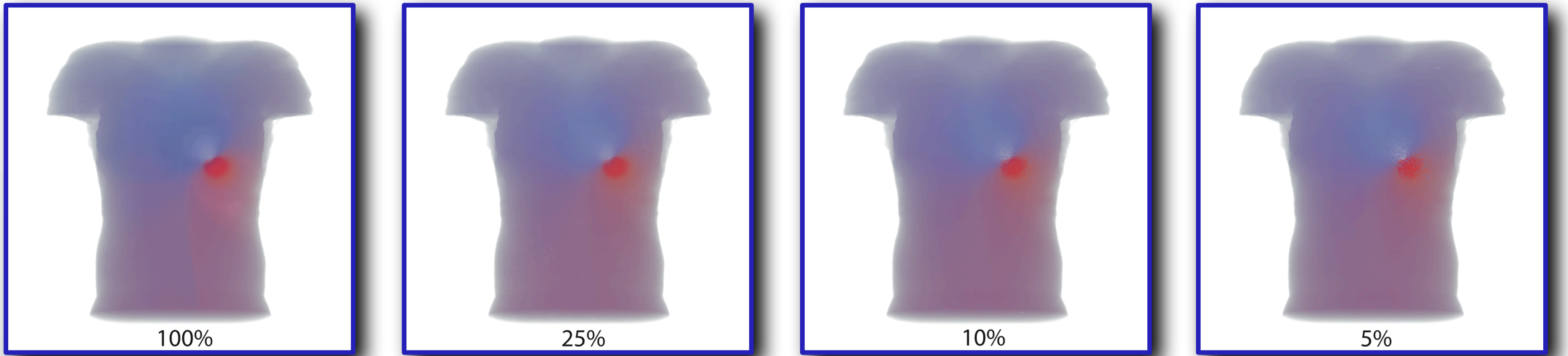
- Hardware-Assisted Visibility Sorting
  - Sort in both object-space and image-space



[Callahan et al. 2005]
http://havs.sourceforge.net and vtk/ParaView

Background

- Dynamic Level-of-Detail

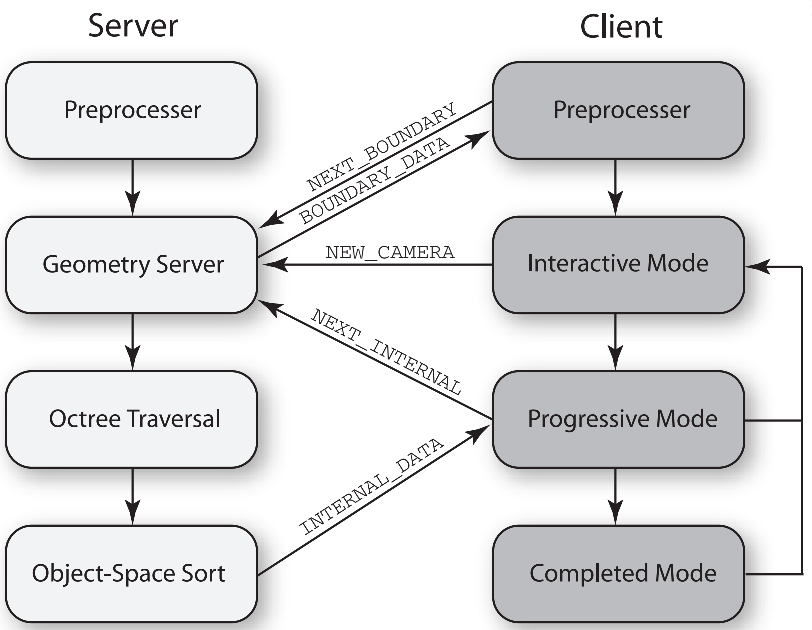


2.0 fps 5.3 fps 10.0 fps 16.1 fps

[Callahan et al. 2005]
http://havs.sourceforge.net and vtk/ParaView

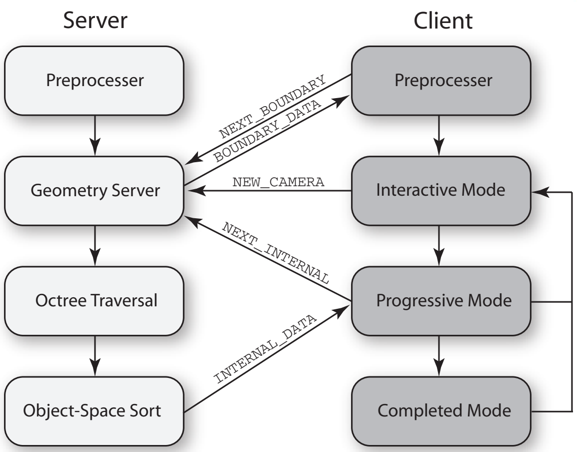
Overview

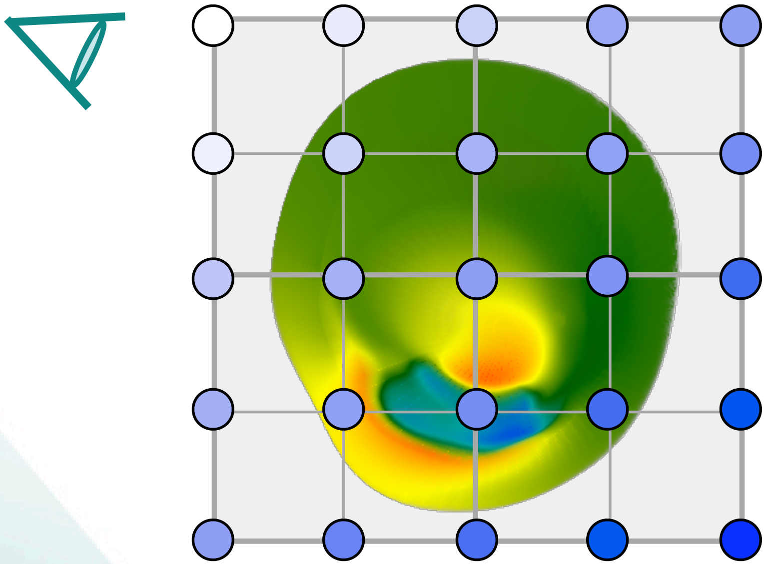
- Server: Processes geometry and transmits triangles in visibility order
- Client: Receives geometry and renders it progressively



The Server

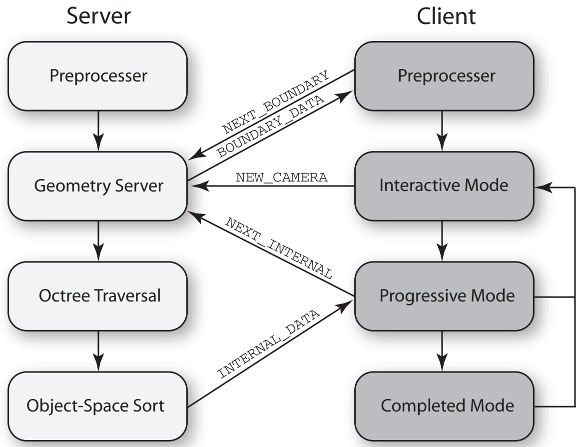
- Preprocess
  - Create min-max octree
- Geometry Server
- Octree Traversal
- Object-Space Sort

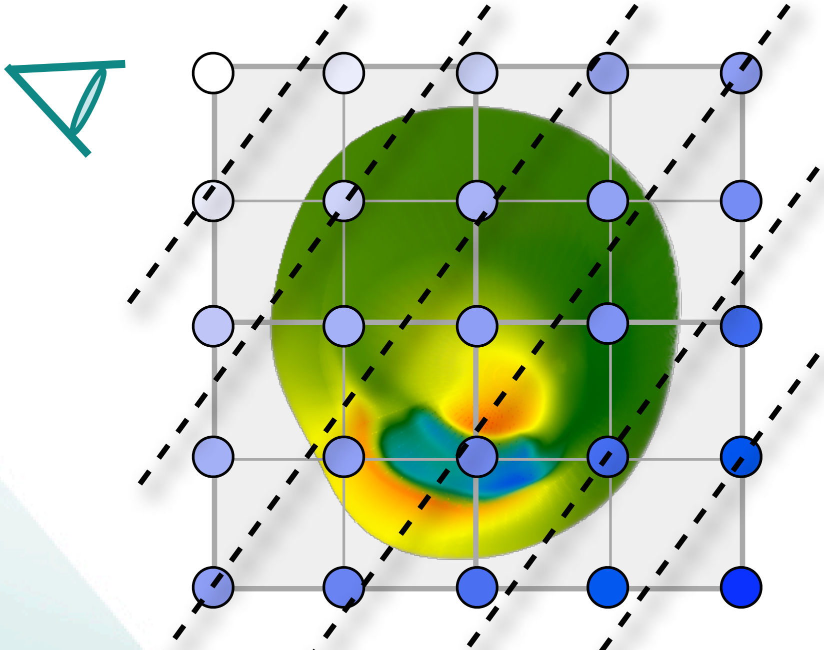




The Server

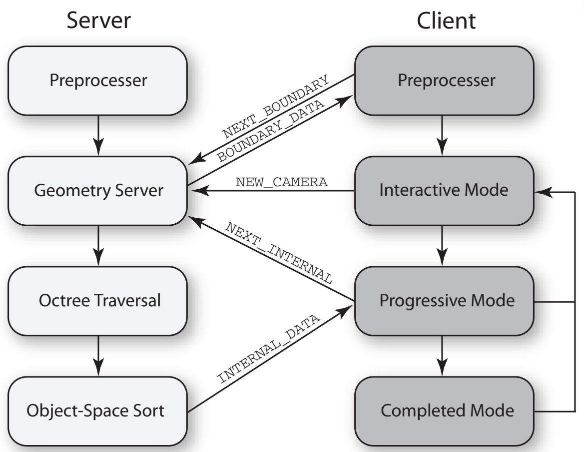
- Preprocess
- Geometry Server
  - Calculate depth range
- Octree Traversal
- Object-Space Sort

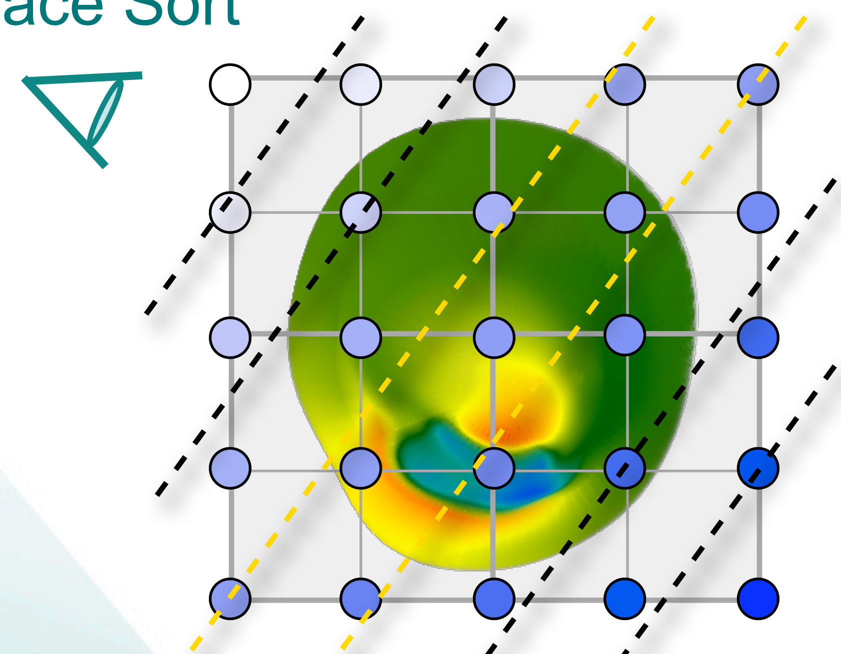




The Server

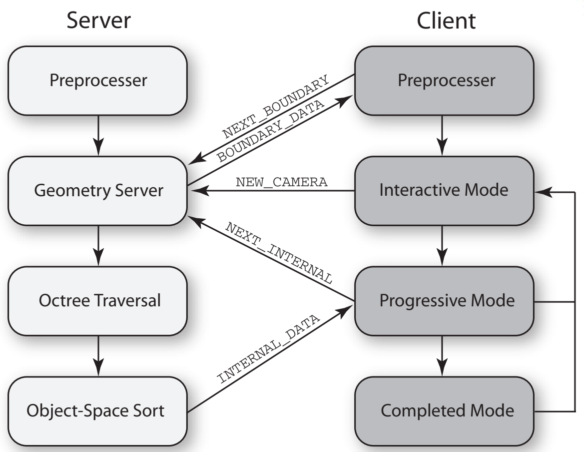
- Preprocess
- Geometry Server
- Octree Traversal
  - Cull range geometry
  - Frustum cull geometry
- Object-Space Sort





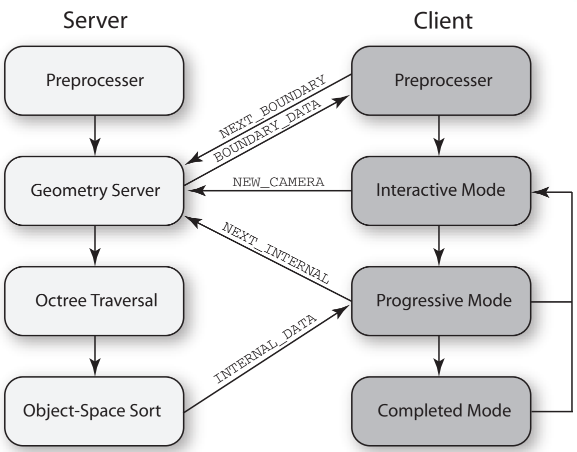
The Server

- Preprocess
- Geometry Server
- Octree Traversal
- Object-Space Sort
  - Sort geometry by centroid
  - Compress and send



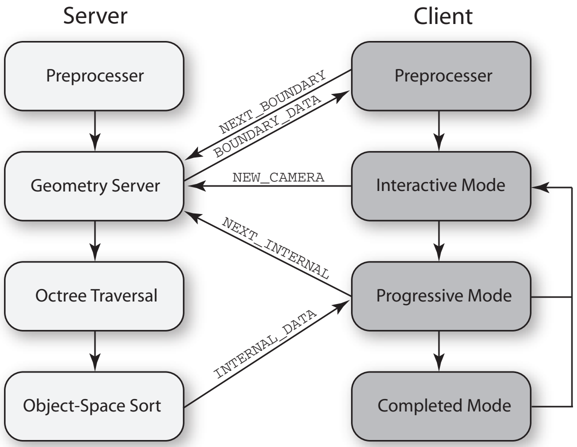
The Client

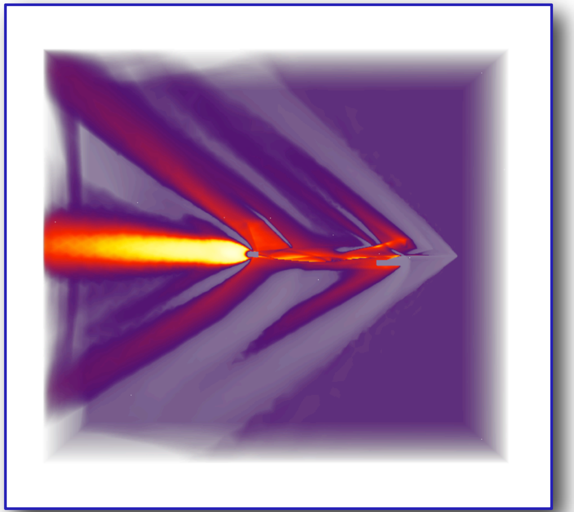
- Preprocess
  - Get boundary geometry from server
  - Build pre-integration table
- Interactive Mode
- Progressive Mode
- Completed Mode



The Client

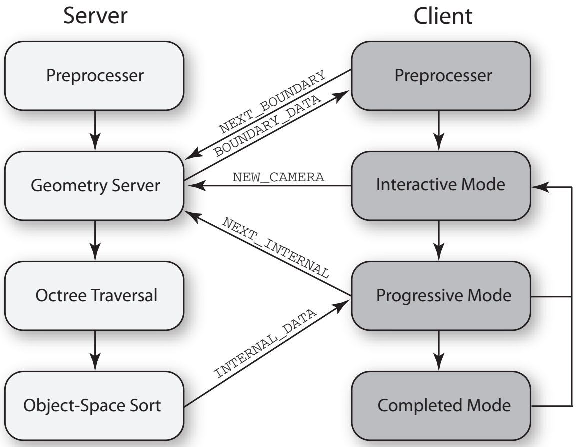
- Preprocess
- Interactive Mode
  - Volume render the boundary geometry
  - Keep the back boundary fragments
- Progressive Mode
- Completed Mode

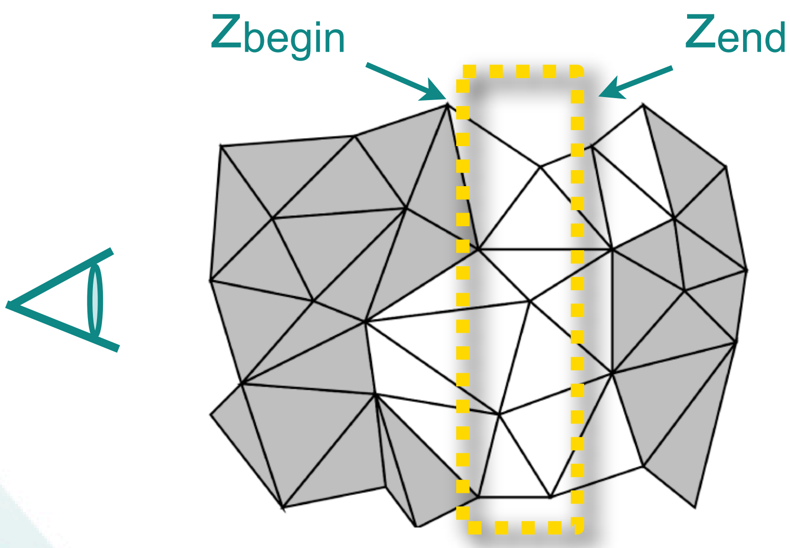




The Client

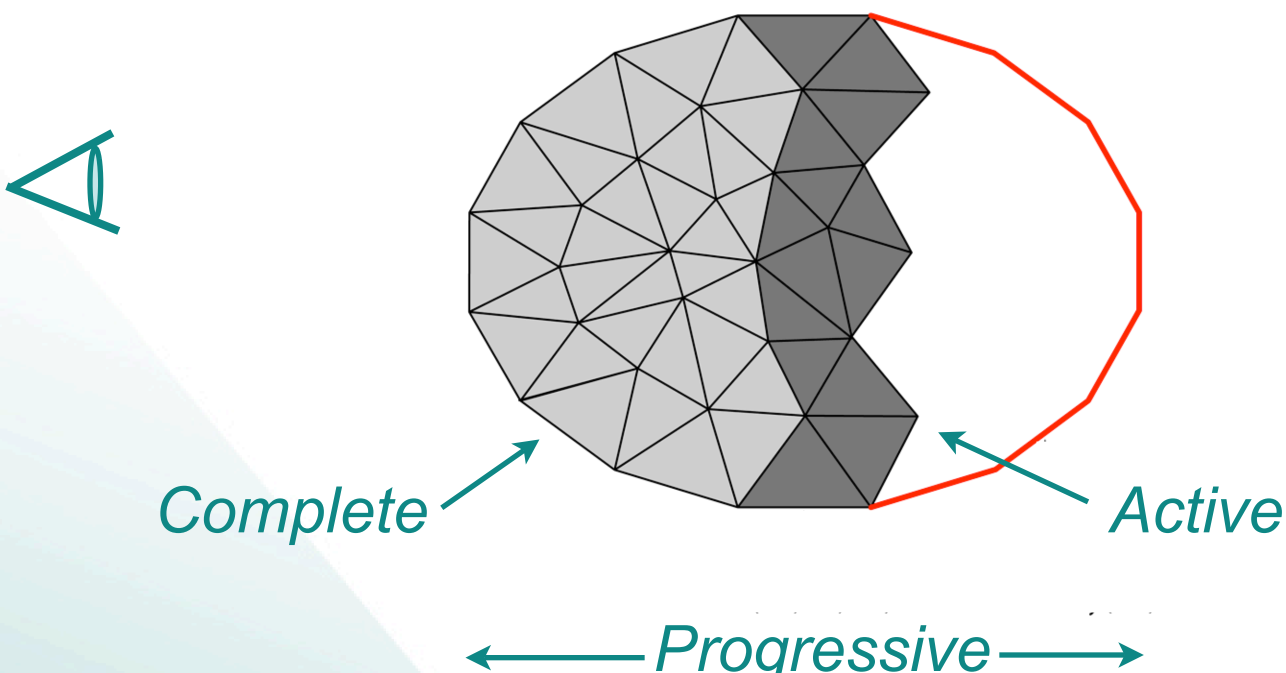
- Preprocess
- Interactive Mode
- Progressive Mode
  - Render range of geometry
  - Display progressive image
- Completed Mode





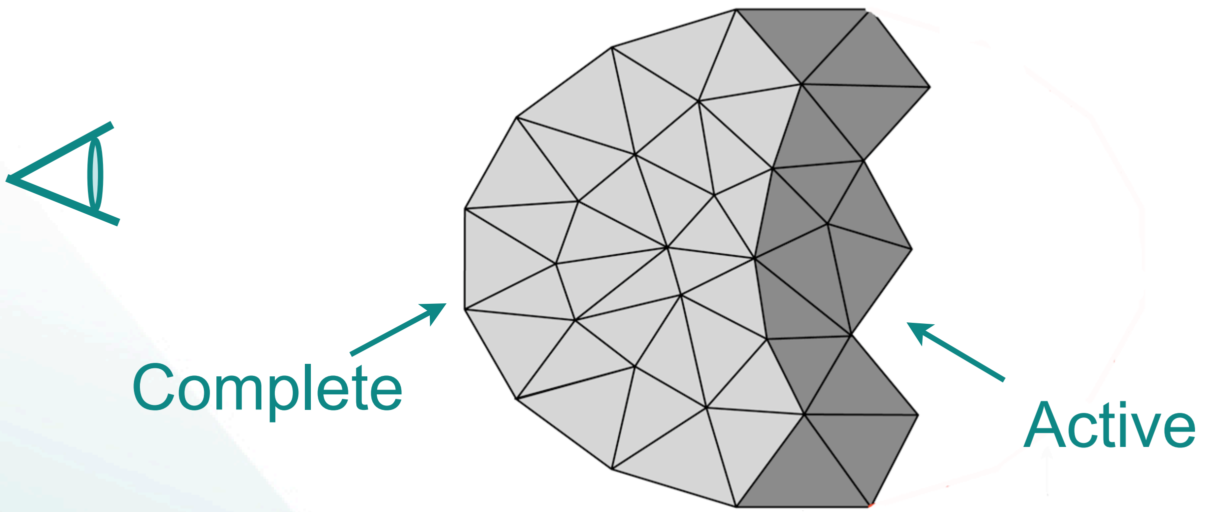
Progressive Mode

- Use three buffers to render progressive image
  - *Complete*: finished volume rendering
  - *Active*: temporary storage of *k* fragments
  - *Progressive*: *Complete* blended with approximation



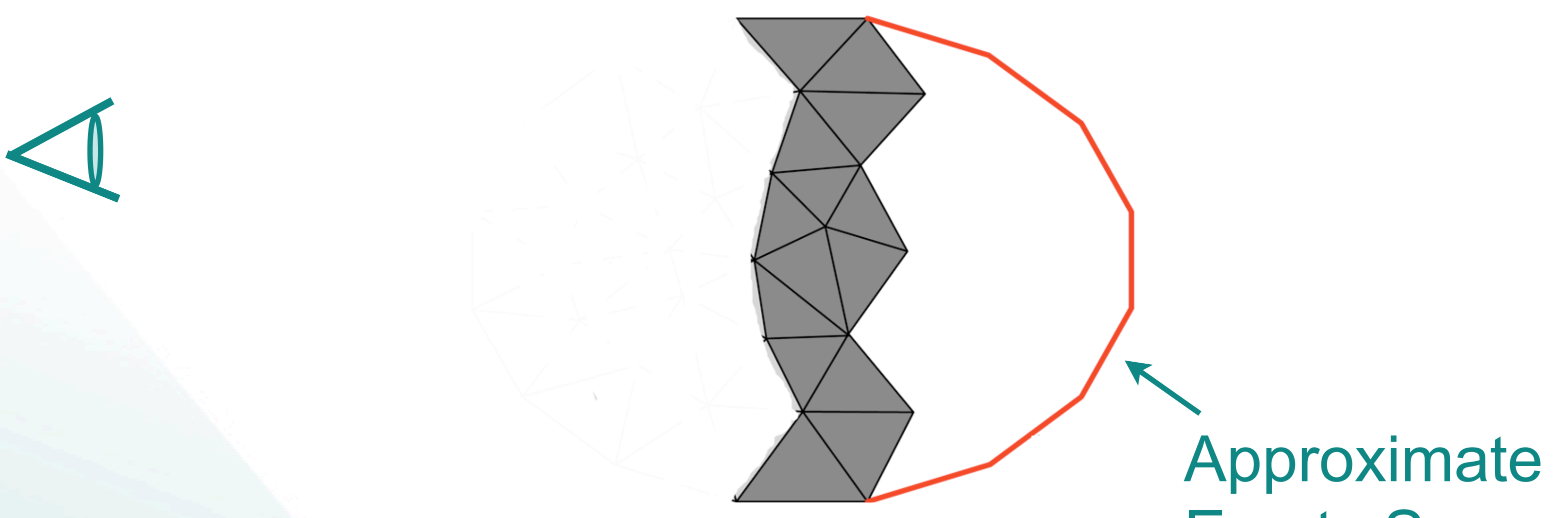
Progressive Mode

- Pass 1:
  - Render geometry into *Active* buffer
  - Composite overflow fragments into *Complete* buffer.



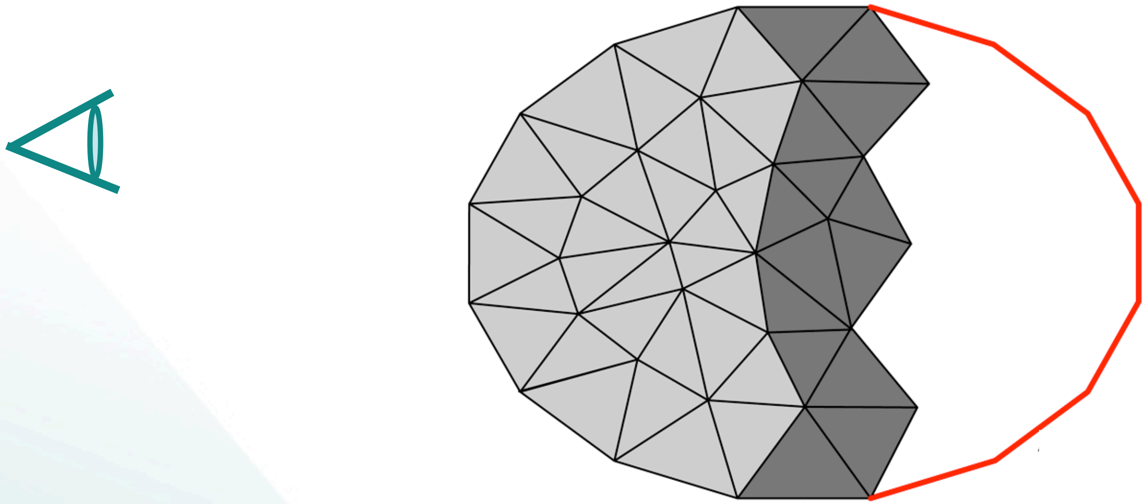
Progressive Mode

- Pass 2:
  - Render empty space into *Progressive* buffer using *Active* buffer and back boundary fragments



Progressive Mode

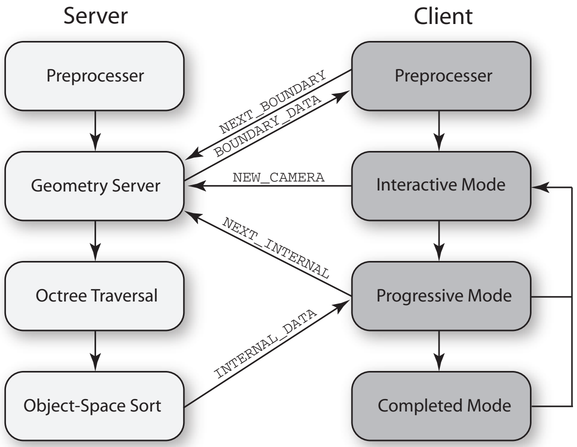
- Pass 3:
  - Composite *Complete* buffer into *Progressive* buffer
  - Display *Progressive* buffer
  - Keep *Complete* and *Active* buffers for next progressive step

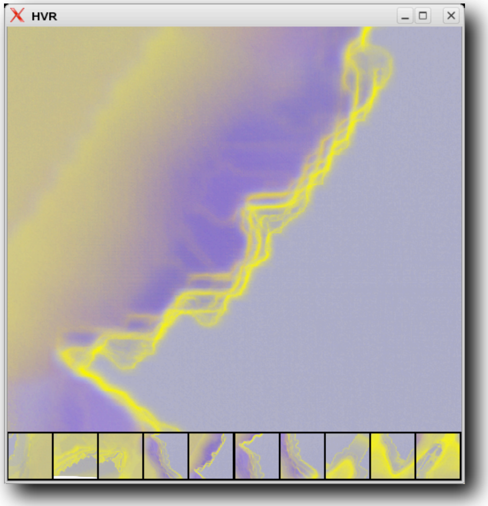


Approximate = Complete + Approximate

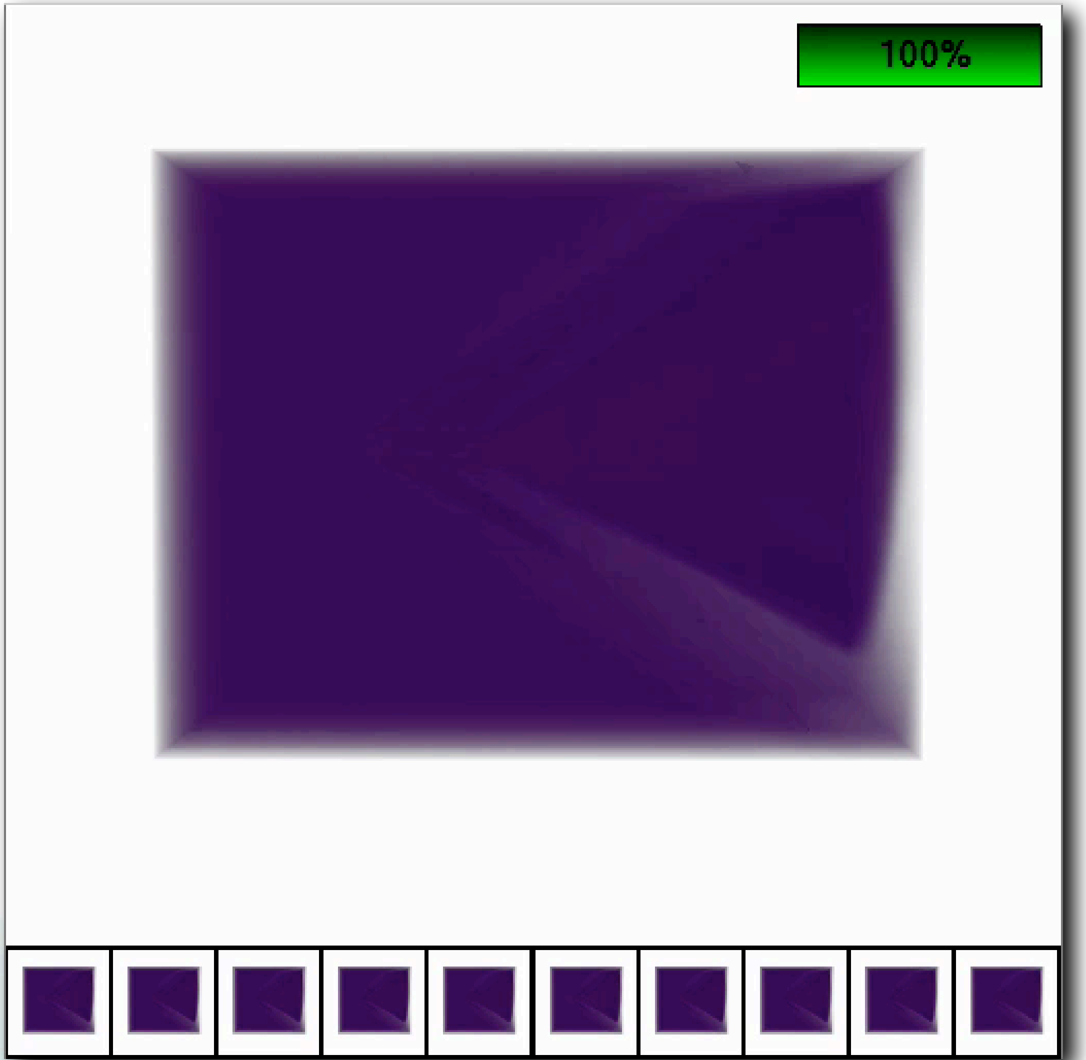
Overview

- Preprocess
- Interactive Mode
- Progressive Mode
- Completed Mode
  - Composite *Active* buffer into *Complete* buffer
  - Display and store *Complete* buffer

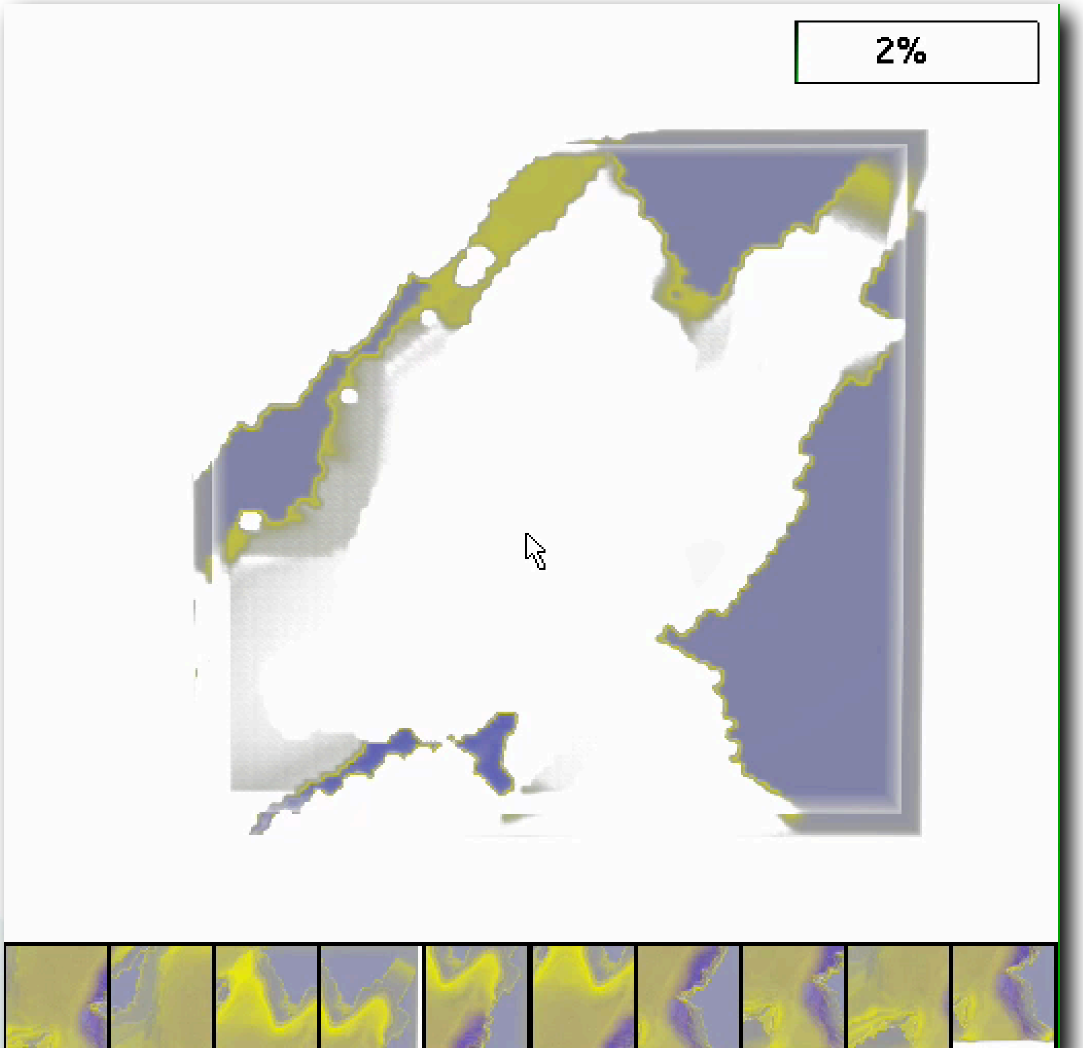




Results

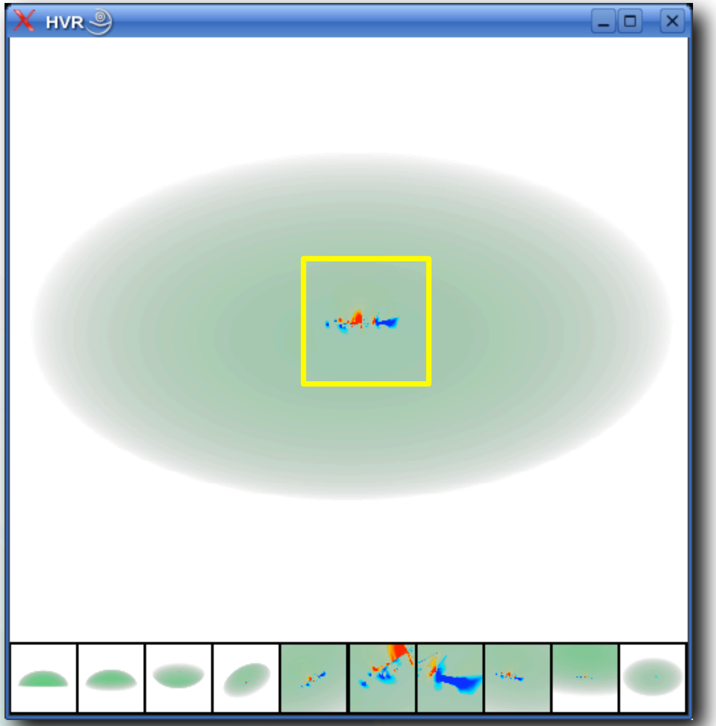


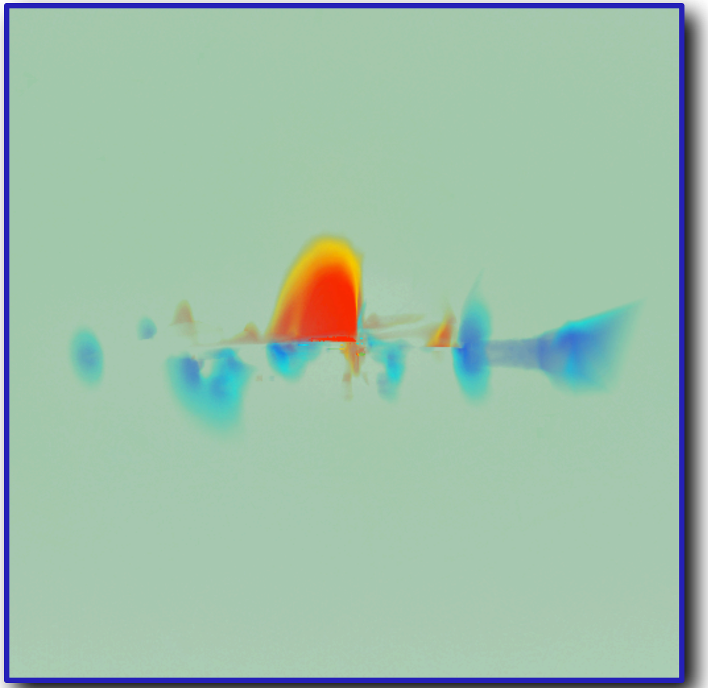
Results



Considerations

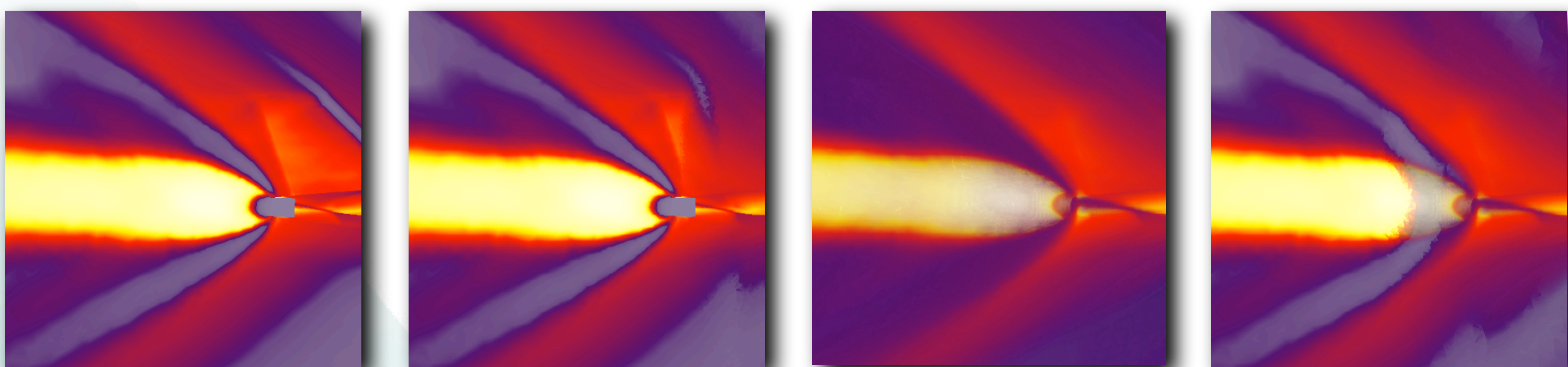
- The network
- Transfer functions
- Other interaction methods





Conclusion

- Remote visualization of large unstructured grids
- Progressions converge to full-quality renderings
- Allows interactive exploration of large datasets
- Future Work:
  - Cutting planes
  - Stream compression
  - Time-varying data



Acknowledgments

- Carlos Scheidegger, Huy Vo
- Datasets
  - Neely and Batina (NASA)
  - O'Hallaran and Shewchuck (CMU)
- Funding
  - DOE
  - IBM
  - SNL
  - LLNL
  - ARO
  - University of Utah