



Application Requirements for Viz

Andrew Siegel
University of Chicago
Argonne National Laboratory



Standard viz story

- ◆ Interested in simulations that benefit from full use of machine
 - O[10Tb] memory
 - O[10 TF/s] proc speed
 - O[1 Pb] storage
- ◆ Do not simply want statistical properties
- ◆ Want to “see” full data in two ways
 - Interactively
 - Steering



Requirements of *Flash*

- ◆ Flash target simulations:
 - AMR grid sizes $\sim 4096^3$
 - Simulation variables ~ 20
 - Saved timesteps ~ 100
 - RAM $\sim 10\text{Tb}$
 - Floppage – 10TF/s
 - Storage – 1Pb
 - Single output file – 100Gb
 - Total viz output – $10\text{-}100\text{Tb}$
 - 3d – full volume rendering



Main point

- ◆ We need robust tools to look at big datasets



Local Data Transfer

- ◆ Typical strategy: transfer data locally
- ◆ Advantages
 - simple: can use proven viz software
 - fast (once you have data)
 - no portability issues



Local Data Transfer

◆ Disadvantages

- 1Tb/day transfer
- need ample local disk
- need ample RAM
- need smart hierarchical viz tool (Porter)
- Particularly bad for group projects



Client/Server System

- ◆ Better strategy: leave data on server
 - back-end rendering software on dedicated procs
 - front-end on client
 - update display as efficiently as possible



Client/Server System

◆ Advantages

- minimal amount of data transferred
- leverage memory, proc speed of server
- do not duplicate data

◆ Disadvantages

- potentially very slow
- very hard to build robustly/portably
- hard to make user-friendly
- existing commercial tools expensive, not redistributable, often not efficient at volume rendering



Runtime vis

◆ Runtime visualization

- Much easier problem than interactive
- Strategy: dump rendered images on server as simulation progresses
- Calls to viz library from within code
- viz library shares memory with code – standardized data layout?



Runtime vis

◆ Runtime vis in Flash

- built off of pvtk
- Still, portability problems
- can reduce overall performance of simulation



Common file formats/datatypes

- ◆ Standardize i/o format across related disciplines
- ◆ Facilitates use of different viz tools with no extra work
- ◆ Improves performance – no copying
- ◆ Colella CCA efforts in this direction for AMR based on HDF5
- ◆ Similar effort for non-AMR?



Current Flash strategy

- ◆ Runtime Viz
 - GD
 - PVTk (not distributed, prototyped)
- ◆ Interactive viz
 - FlashViz (ANL)
 - XFlash (homemade, mainly 2d)
- ◆ Wrappers for
 - ChomboVis, Visamrai, TecPlot, Vis, homemade IDL tools, ...



Key Issues

- ◆ System balance is key issue
- ◆ Hierarchical tools
- ◆ Data reduction, compression
- ◆ Research software vs. commercial software
 - support, portability, ease-of-use, etc. for distributed tools