Why You Don’t Want to do In Situ Visualization, and Why You Have To

ISC Workshop On In Situ Visualization 2017

Kenneth Moreland Sandia National Laboratories

June 22, 2017
Acknowledgements

- This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, under Award Numbers 10-014707, 12-015215, and 14-017566.

- This research was supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of two U.S. Department of Energy organizations (Office of Science and the National Nuclear Security Administration) responsible for the planning and preparation of a capable exascale ecosystem, including software, applications, hardware, advanced system engineering, and early testbed platforms, in support of the nation’s exascale computing imperative.

- Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy’s National Nuclear Security Administration under contract DE-NA-0003525.

- Thanks to many, many partners in labs, universities, and industry.
What Does In Situ Visualization Mean?

Co-processing  "Strict" In Situ
In Transit
Simulation Steering
Integrated Problem Solving Environments
On-Line Analysis
Tightly/Loosely Couples
In Situ Visualization is visualization that necessarily starts before the data producer finishes.
In Situ Visualization is visualization that *necessarily* starts before the data producer finishes.

Post Hoc Visualization is visualization that can start at some arbitrary time after the data are produced.
Reasons Why We Don’t All Do In Situ Visualization All the Time
Loss of Interactivity
Expressing the Visualization
Expressing the Visualization

```python
# Create a view of the ball impacting the brick
ballblock = ExtractCTHBlocks(ctsource)
b(ballblock.MaterialFractionThreshold = 0.5)
b(ballblock.SelectMaterialFractionArrays = ["VOLM1", "VOLM2"]
b(ballblock.SelectMassArrays = ["M1", "M2"]

# Reflect the simulation result about the simulation axis of symmetry
normBall = GenerateSurfaceNormals(ballblock)
normBall.FeatureAngle = 90
normBall.Splitting = 1
normBall.PieceInvariant = 1
reflBall = Reflect(normBall)
reflBall.Center = 0.0
reftBall.Plane = "x"

# Function to make the appropriate colorbar
def MakeGoldRedLT():
    lt = servermanager.rendering.PVLookupTable()
    servermanager.Register(lt)
    lt.RGBPoints = [1, 0.56, 0.23, 0.23, 0, 0.9568, 0.6928, 0.3294]
    lt.ColorSpace = "RGB"
    return lt

# Function to initialize the display
def SetupDisplay(proxy, lt):
    SetDisplayProperties(proxy, SuppressLOD = 1)
    SetDisplayProperties(proxy, StaticMode = 1)
    SetDisplayProperties(proxy, LookupTable = lt)
    SetDisplayProperties(proxy, Specular = 0.1)
    SetDisplayProperties(proxy, SpecularPower = 40)
    SetDisplayProperties(proxy, SpecularColor = [1.0, 1.0, 1.0])

# Setup the initial display
```
Using Up Resources

Error: Out of Memory.
More Complexity = Less Robustness

dracut: FATAL: Initial SELinux policy load failed. Machine in enforcing mode. To disable selinux, add selinux=0 to the kernel command line.
dracut: Refusing to continue

Kernel panic - not syncing: Attempted to kill init!
Scaling

![Graph showing scaling of Rate (Blocks/Second) vs. Number of Cores]

- Ideal
- 220k blocks
- 1.5m blocks
- 33k blocks

![Graph showing Time (sec) vs. Number of Processes]

- 512
- 1024
- 2048
- 4096
- 8192
- 16384
- 32768
- 65536

Collect
Composite
Should We do In Situ Visualization at All?
In Situ Visualization

Pros
- Batch/Automated Processing
- Early Access to Data
- Access to More Data

Cons
- Loss of Interactivity
- Hard to Express the Visualization
- Difficult Collaborations of Code Teams
- Extra Burden of Resources
- Less Robust
- Requires More Scaling
Post Hoc Batch/Automated Processing

Post-Processing V&V Level II ASC Milestone

- FY07
  - Scripting for automated post processing
  - User defined functions at run time
  - Scalability
- FY08
  - Fragment identification and characterization

Python Scripting, Server Side

Presented at DOE Computer Graphics Forum 2008
## In Situ Visualization

### Pros
- Batch/Automated Processing (crossed out)
- Early Access to Data
- Access to More Data

### Cons
- Loss of Interactivity
- Hard to Express the Visualization
- Difficult Collaborations of Code Teams
- Extra Burden of Resources
- Less Robust
- Requires More Scaling
In Situ Visualization is visualization that necessarily starts before the data producer finishes.

Post Hoc Visualization is visualization that can start at some arbitrary time after the data are produced.
I suggest you ...

--- General

**307**

Voles

Vote

**Reload-Button**

It would be nice to have a reload-button for time-dependend datasets. So you don't have to apply all your filters again if the simulation is evolved in time.

R. Prignitz shared this idea  · Sep 14, 2009  · Flag idea as inappropriate...

**COMPLETED**  · Utkarsh Ayachit (Admin, paraview) responded  · Sep 7, 2016

This feature is now available in the development repository and will be included in the next release, ParaView 5.2.

See the merge-request (now merged) for details: https://gitlab.kitware.com/paraview/paraview/merge_requests/1000

For details see https://blog.kitware.com/refreshingreloading-files-in-paraview/

Show previous admin responses (2)
In Situ Visualization

Pros
- Batch/Automated Processing
- Early Access to Data
- Access to More Data

Cons
- Loss of Interactivity
- Hard to Express the Visualization
- Difficult Collaborations of Code Teams
- Extra Burden of Resources
- Less Robust
- Requires More Scaling
Why We \textit{Have} to do In Situ Visualization
Node Memory
4.5 PB/s

Computation
125 PB/s
Computation
125 PB/s

Node Memory
4.5 PB/s

Interconnect
24 TB/s

(Largest Cross-Sectional Bandwidth)
- Computation: 125 PB/s
- Node Memory: 4.5 PB/s
- Interconnect: 24 TB/s
- (Largest Cross-Sectional Bandwidth)
- Storage: 1.4 TB/s
Computation 125 PB/s

Node Memory 4.5 PB/s

Interconnect 24 TB/s

Storage 1.4 TB/s

> 3000x difference
Summit’s Dirty Little Secret

- Summit is the next generation supercomputer replacing Titan
- Summit is expected to have at least 5x improvement in FLOPS
- Summit is expected to have *no* improvement in file storage bandwidth
In Situ Visualization

Pros
- Batch/Automated Processing
- Early Access to Data
- Access to More Data

Cons
- Loss of Interactivity
- Hard to Express the Visualization
- Difficult Collaborations of Code Teams
- Extra Burden of Resources
- Less Robust
- Requires More Scaling

We need this when we cannot save enough to disk.
Addressing Shortcomings of In Situ
Loss of Interactivity
Write Smarter, Not Harder

- Use analysis to select important data
  - Entropy, feature detection, topological methods, machine learning
- Transform data for better information extraction
  - Compression, Lagrangian flow representation


Expressing the Visualization

```python
# Create a view of the ball impacting the brick
ballblock = ExtractCHFragments (cthsource)
bballblock.MaterialFractionThreshold = 0.5
ballblock.SelectMaterialFractionArrays = ["VOLM1", "VOLM2"]
bballblock.SelectMassArrays = ["M1", "M2"]

# Reflect the simulation result about the simulation axis of symmetry
normBall = GenerateSurfaceNormals (ballblock)
normBall.FeatureAngle = 90
normBall.Splitting = 1
normBall.PieceInvariant = 1
reflBall = Reflect (normBall)
reflBall.Center = 0.0
reflBall.Plane = "x"

# Function to make the appropriate colorbar
def MakeGoldRedLT()
    lt = servermanager.rendering.PVLookupTable ()
    servermanager.Register(lt)
    lt.RGBPoints = [1, 0.56, 0.23, 0.23, 0, 0.9568, 0.6928, 0.3294]
    lt.ColorSpace = "RGB"
    return lt

# Function to initialize the display
def SetupDisplay(proxy, lt):
    SetDisplayProperties (proxy, SuppressLOD = 1)
    SetDisplayProperties (proxy, StaticMode = 1)
    SetDisplayProperties (proxy, LookupTable = lt)
    SetDisplayProperties (proxy, Specular = 0.1)
    SetDisplayProperties (proxy, SpecularPower = 40)
    SetDisplayProperties (proxy, SpecularColor = [1.0, 1.0, 1.0])

# Setup the initial display
```
Cramming Codes Together
Using Up Resources

Error: Out of Memory.
More Complexity = Less Robustness

grsec: use of CAP_SYS_ADMIN in chroot denied for /sysroot/sbin/load_policy[load_policy:260] uid/egid:0/0
grsec: use of CAP_SYS_ADMIN in chroot denied for /sysroot/sbin/load_policy[load_policy:260] uid/egid:0/0
grsec: use of CAP_SYS_ADMIN in chroot denied for /sysroot/sbin/load_policy[load_policy:260] uid/egid:0/0
grsec: use of CAP_SYS_ADMIN in chroot denied for /sysroot/sbin/load_policy[load_policy:260] uid/egid:0/0
grsec: use of CAP_SYS_ADMIN in chroot denied for /sysroot/sbin/load_policy[load_policy:260] uid/egid:0/0
grsec: use of CAP_SYS_ADMIN in chroot denied for /sysroot/sbin/load_policy[load_policy:260] uid/egid:0/0
grsec: use of CAP_SYS_ADMIN in chroot denied for /sysroot/sbin/load_policy[load_policy:260] uid/egid:0/0
grsec: use of CAP_SYS_ADMIN in chroot denied for /sysroot/sbin/load_policy[load_policy:260] uid/egid:0/0
dracut: FATAL: Initial SELinux policy load failed. Machine in enforcing mode. To disable selinux, add selinux=0 to the kernel command line.
dracut: Refusing to continue

Kernel panic - not syncing: Attempted to kill init!
Scaling

Graph showing the rate of blocks processed per second against the number of cores for different block sizes. The ideal performance is indicated, with data points for 33k, 220k, and 1.5m blocks. A separate bar chart shows the time taken for different numbers of processes (512 to 66536).
Alternative to Scaling

Simulation

Computation Nodes

Simulation Results

Staging Nodes

Vis

Interactive Vis Controls

Visualization Results

Vis Client
Conclusions

- In Situ takes away lots of beloved features of visualization
  - This has limited its adoption to specific use cases for almost 50 years
- The resurgence of interest (why we are here today) is because there is a pressing need in HPC
- File I/O speed has been losing ground since forever
  - From a practical standpoint, we are hitting resource limits that force us to move to in situ
- Understand why you are doing in situ and what you are giving up to be successful
Acknowledgements

- This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, under Award Numbers 10-014707, 12-015215, and 14-017566.

- This research was supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of two U.S. Department of Energy organizations (Office of Science and the National Nuclear Security Administration) responsible for the planning and preparation of a capable exascale ecosystem, including software, applications, hardware, advanced system engineering, and early testbed platforms, in support of the nation’s exascale computing imperative.

- Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy’s National Nuclear Security Administration under contract DE-NA-0003525.

- Thanks to many, many partners in labs, universities, and industry.