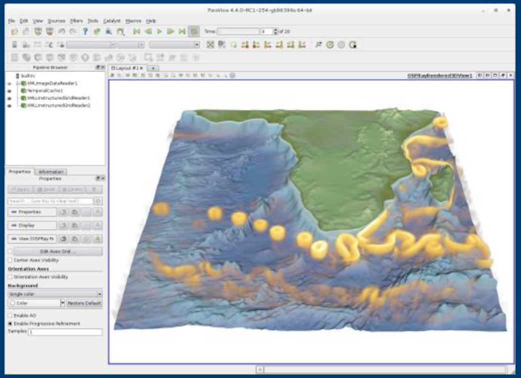
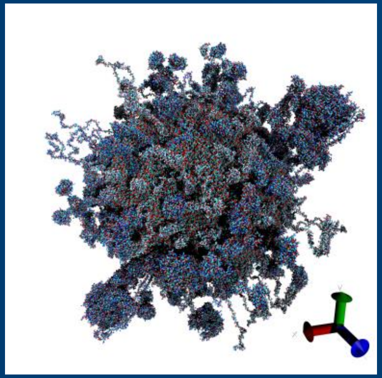
OSPRay – A CPU Ray Tracing Framework for Scientific Visualization Rendering

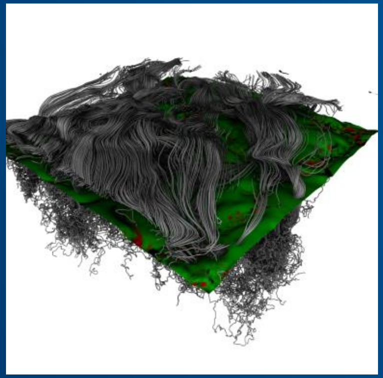
Ingo Wald [$]   Gregory P Johnson [$]   Jefferson Amstutz [$]   Carson Brownlee [+ $]

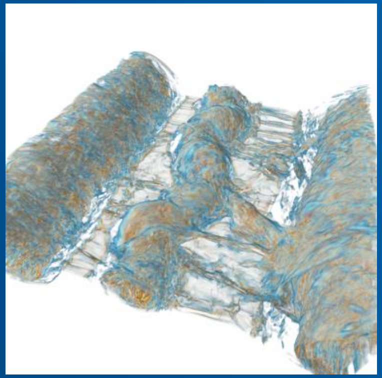
*Aaron Knoll [# *]   Jim Jeffers [$]   Johannes Guenther [$]   Paul Navratil [+]*

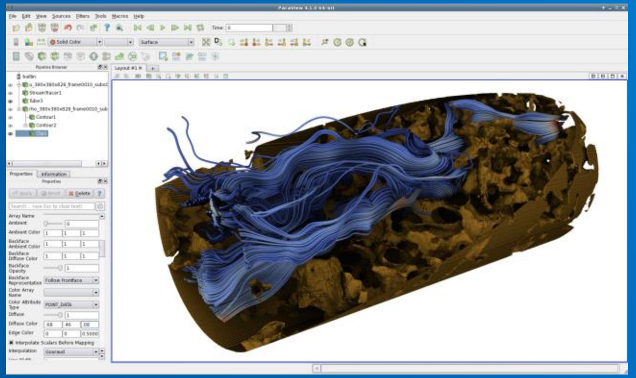
[$]Intel   [*]SCI   [+]TACC   [#]ANL











Motivation

What I heard more than once (this week, too!):

Rendering is a solved Problem

Either: “rendering is not the real problem in vis, anyway”

Or: “just plug in your latest GPU – everybody has one, anyway – and done!”

A solved problem? Well, maybe … *except* …

A solved problem? Well, maybe … *except* …

a) Not everybody does have a GPU

- Yes, your workstation does …
- … but for large data, you can't move it there any more

→ Want to ideally render *right on the compute node(s)*

A solved problem? Well, maybe ... *except* ...

a) Not everybody does have a GPU

Rank	Site	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	National Supercomputing Center in Wuxi China	**Sunway TaihuLight** - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway NRCPC	10,649,600	93,014.6	125,435.9	15,371
2	National Super Computer Center in Guangzhou China	**Tianhe-2 (MilkyWay-2)** - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P NUDT	3,120,000	33,862.7	54,902.4	17,808
3	DOE/SC/Oak Ridge National Laboratory United States	**Titan** - Cray XK7 , Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x Cray Inc.	560,640	17,590.0	27,112.5	8,209
4	DOE/NNSA/LLNL United States	**Sequoia** - BlueGene/Q, Power BQC 16C 1.60 GHz, Custom IBM	1,572,864	17,173.2	20,132.7	7,890
5	RIKEN Advanced Institute for Computational Science (AICS) Japan	K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect Fujitsu	705,024	10,510.0	11,280.4	12,660
6	DOE/SC/Argonne National Laboratory United States	**Mira** - BlueGene/Q, Power BQC 16C 1.60GHz, Custom IBM	786,432	8,586.6	10,066.3	3,945
7	DOE/NNSA/LANL/SNL United States	**Trinity** - Cray XC40, Xeon E5-2698v3 16C 2.3GHz, Aries interconnect Cray Inc.	301,056	8,100.9	11,078.9	
8	Swiss National Supercomputing Centre (CSCS) Switzerland	**Piz Daint** - Cray XC30, Xeon E5-2670 8C 2.600GHz, Aries interconnect , NVIDIA K20x Cray Inc.	115,984	6,271.0	7,788.9	2,325
9	HLRS - Höchstleistungsrechenzentrum Stuttgart Germany	**Hazel Hen** - Cray XC40, Xeon E5-2680v3 12C 2.5GHz, Aries interconnect Cray Inc.	185,088	5,640.2	7,403.5	
10	King Abdullah University of Science and Technology Saudi Arabia	**Shaheen II** - Cray XC40, Xeon E5-2698v3 16C 2.3GHz, Aries interconnect Cray Inc.	196,608	5,537.0	7,235.2	2,834

A solved problem? Well, maybe … *except* …

a) Not everybody does have a GPU

Rank	Site	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	National Supercomputing Center in Wuxi China	**Sunway TaihuLight** - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway NRCPC	10,649,600	93,014.6	125,435.9	15,371
2	National Super Computer Center in Guangzhou China	**Tianhe-2 (MilkyWay-2)** - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P NUDT	3,120,000	33,862.7	54,902.4	17,808
3	DOE/SC/Oak Ridge National Laboratory United States	**Titan** - Cray XK7 , Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x Cray Inc.	560,640	17,590.0	27,112.5	8,209
4	DOE/NNSA/LLNL United States	**Sequoia** - BlueGene/Q, Power BQC 16C 1.60 GHz, Custom IBM	1,572,864	17,173.2	20,132.7	7,890
5	RIKEN Advanced Institute for Computational Science (AICS) Japan	K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect Fujitsu	705,024	10,510.0	11,280.4	12,660
6	DOE/SC/Argonne National Laboratory United States	**Mira** - BlueGene/Q, Power BQC 16C 1.60GHz, Custom IBM	786,432	8,586.6	10,066.3	3,945
7	DOE/NNSA/LANL/SNL United States	**Trinity** - Cray XC40, Xeon E5-2698v3 16C 2.3GHz, Aries interconnect Cray Inc.	301,056	8,100.9	11,078.9	
8	Swiss National Supercomputing Centre (CSCS) Switzerland	**Piz Daint** - Cray XC30, Xeon E5-2670 8C 2.600GHz, Aries interconnect , NVIDIA K20x Cray Inc.	115,984	6,271.0	7,788.9	2,325
9	HLRS - Höchstleistungsrechenzentrum Stuttgart Germany	**Hazel Hen** - Cray XC40, Xeon E5-2680v3 12C 2.5GHz, Aries interconnect Cray Inc.	185,088	5,640.2	7,403.5	
10	King Abdullah University of Science and Technology Saudi Arabia	**Shaheen II** - Cray XC40, Xeon E5-2698v3 16C 2.3GHz, Aries interconnect Cray Inc.	196,608	5,537.0	7,235.2	2,834

A solved problem? Well, maybe … *except* …

a) Not everybody does have a GPU

- Yes, your workstation does …
- … but for large data, you can't move it there any more

→Want to ideally render *right on the compute node(s)*

- … and most of those are mostly CPU nodes
  (and so will most upcoming systems be: Trinity, Cori, Theta, Stampede 2, …)

A solved problem? Well, maybe ... *except* ...

a) Not everybody does have a GPU

b) Even if you do have some, GPUs have / create issues, too

- Ignore cost, power, admin cost, ...
- Two key problems in part for today's "big data" challenges:
  - On other side of (slow) PCI bus
  - Limited memory (order(s) of magnitude smaller than main memory)

A solved problem? Well, maybe … *except* …

a) Not everybody does have a GPU

b) Even if you do have some, GPUs have / create issues, too

c) OpenGL may not be the best choice (any more), anyway

- why would I want to tessellate spheres/cylinders/etc to gazillions of triangles!?
- why should I have to extract/store/render 100's of M's of tris to render an iso-surface?
- do I really want to render 100's of M's of tris per frame? on a 1M pixel screen?
- why is adding a bit of transparency such a big deal? Or volumes+surfaces!?
- …

A solved problem? Well, maybe … *except* …

a) Not everybody does have a GPU

b) Even if you do have some, GPUs have / create issues, too

c) OpenGL may not be the best choice (any more), anyway

→ Vis rendering might actually benefit from CPU rendering

- Every compute node becomes a vis node (availability, scalability, …)
- Full access to full memory
- In situ …

A solved problem? Well, maybe … *except* …

a) Not everybody does have a GPU

b) Even if you do have some, GPUs have / create issues, too

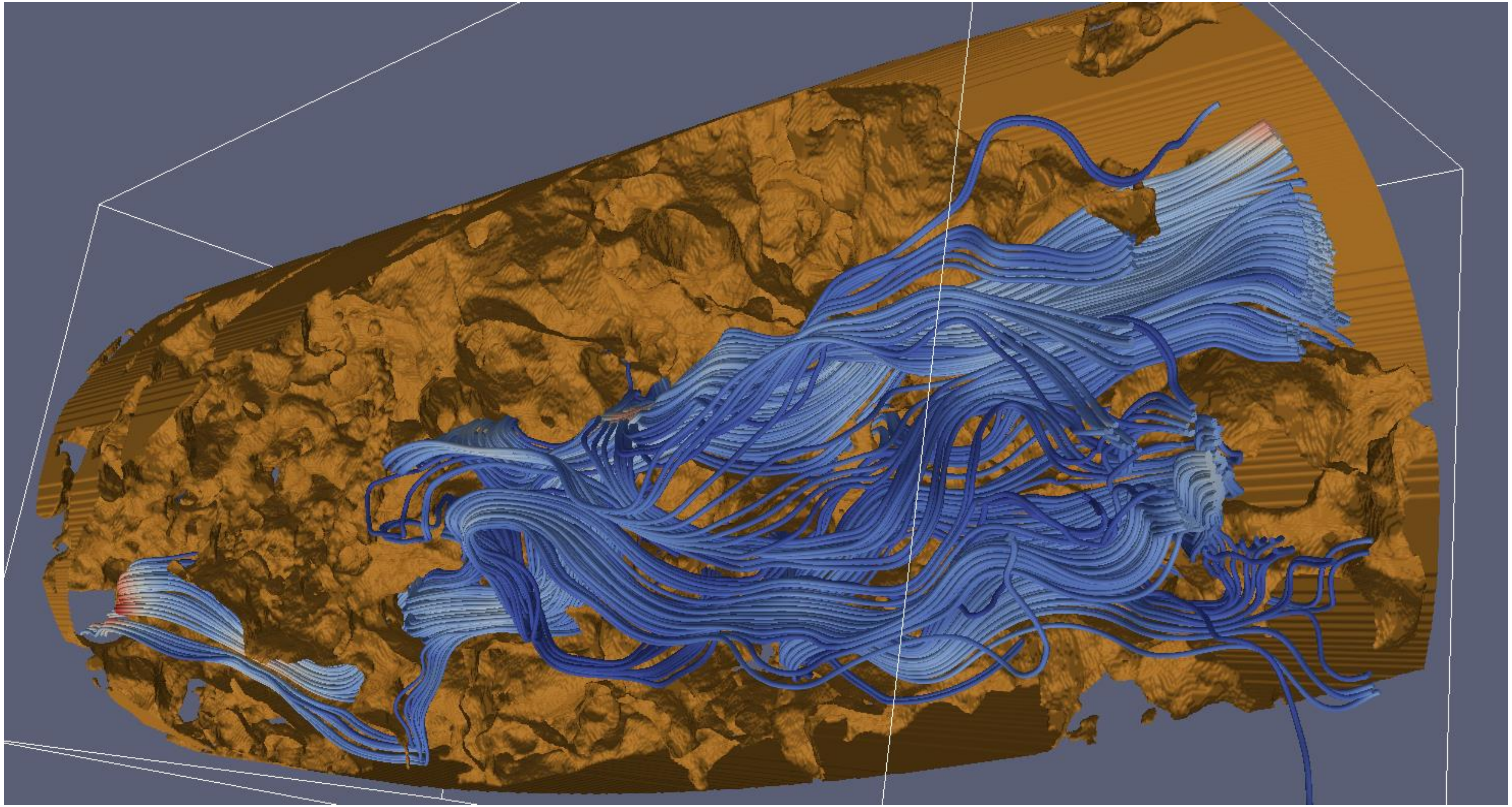
c) OpenGL may not be the best choice (any more), anyway

→ Vis rendering might actually benefit from CPU rendering

→ Vis rendering might actually benefit from *ray tracing*

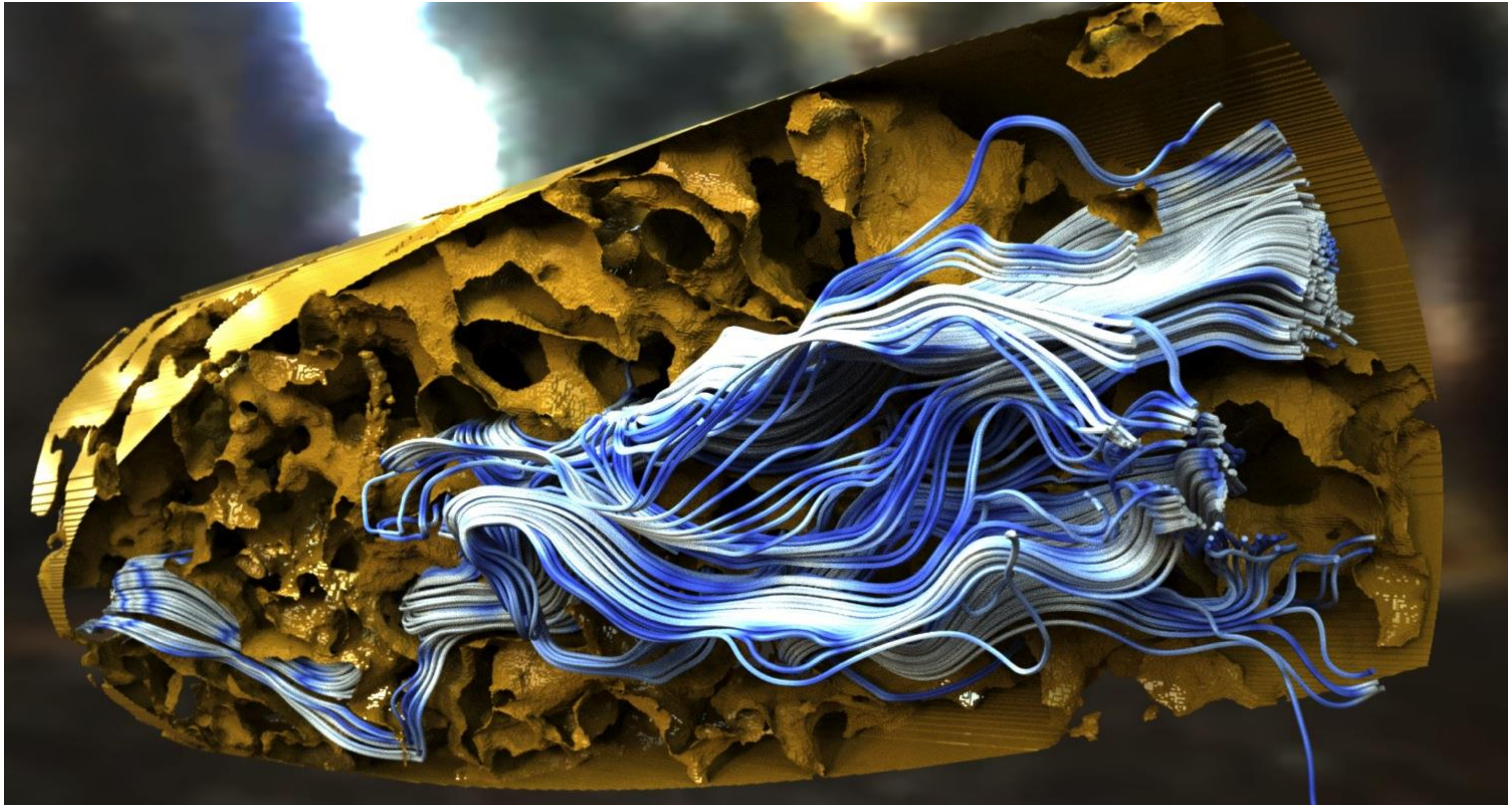
- *Efficiency:* scalability in model size, no need to tessellate, volumes+surfaces, …
- *Effectiveness:* better shading for *more effective* visualizations

Advanced Shading for more *effective* visualization



OpenGL based Visualization (today)

Advanced Shading for more *effective* visualization



Ray Tracing based vis (OSPRay)

A solved problem? Well, maybe … *not*?

In summary, sci-vis could benefit from two things:

a) A fast rendering solution for CPUs

→ "Software Defined Visualization" (could also be rasterization)

b) The option to render with ray tracing

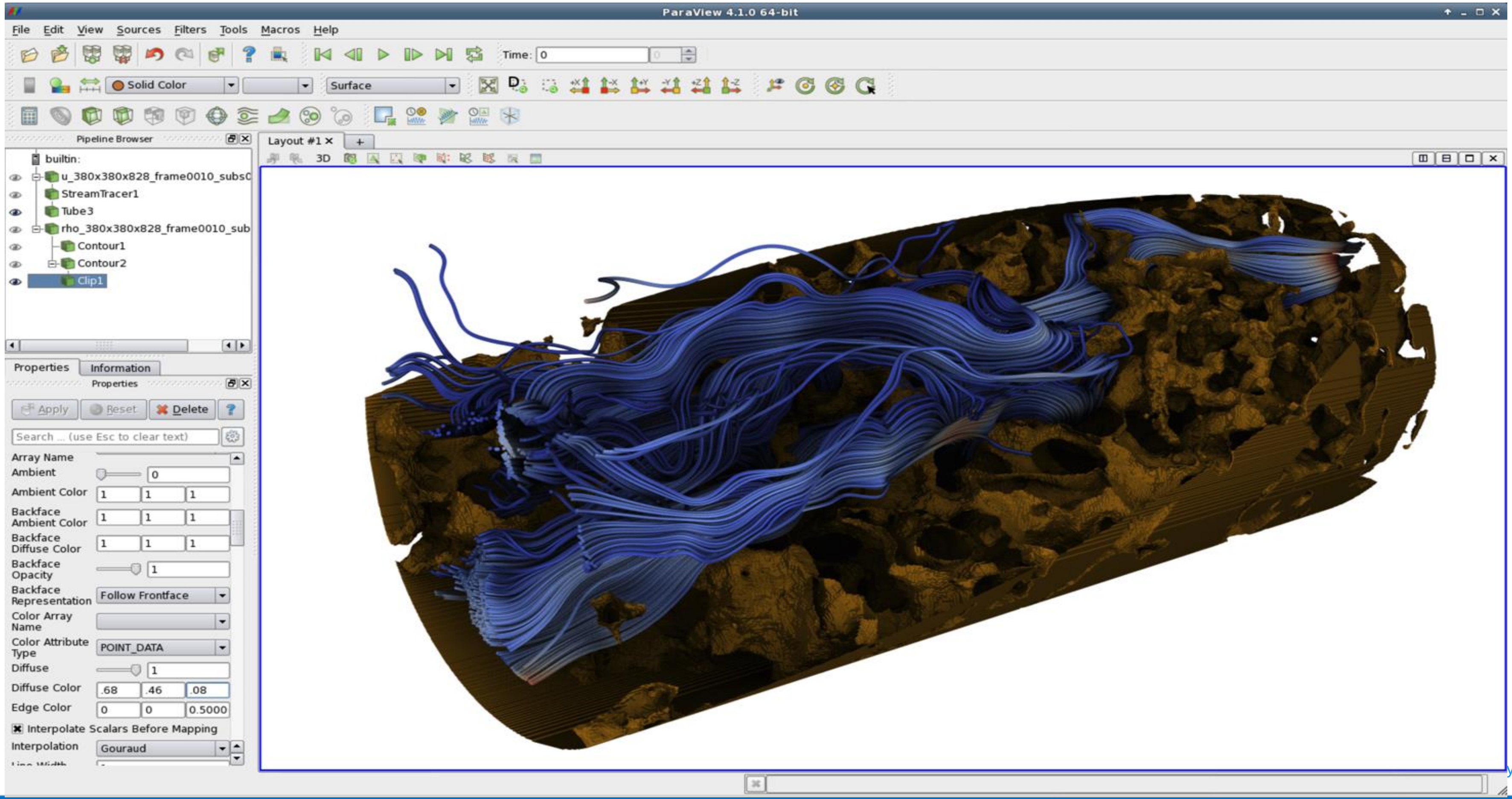
→ "High Fidelity Visualization" (could also be a GPU)

➔ OSPRay: CPU Rendering + Ray Tracing (for Vis)

OSPRay

What is OSPRay?

- OSPRay: “A CPU ray tracing framework/library for scientific visualization rendering”

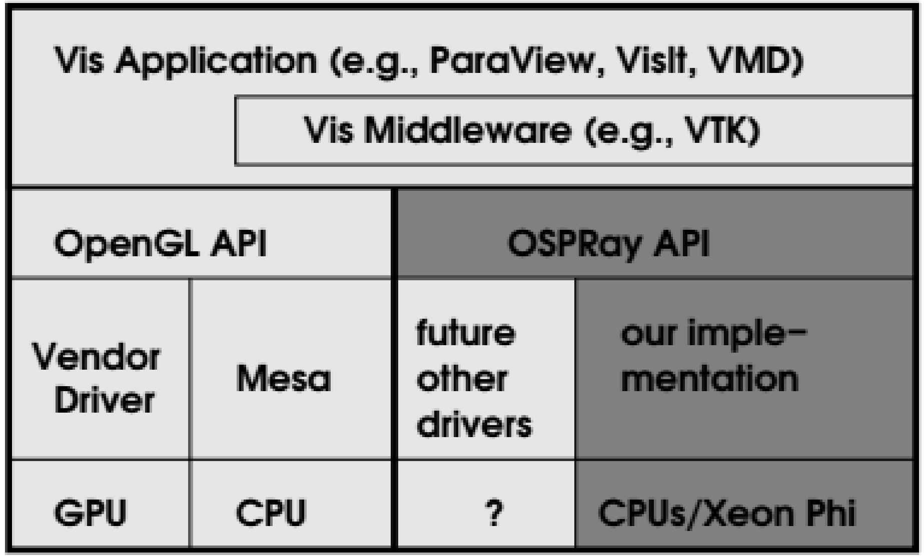


OSPRay Goals

- Offer Compelling CPU rendering solution for Visualization
  - Target upcoming systems such as Stampede 2, Trinity, Cori, Theta, …
- Focus on Visualization (not games, not movies)
  - Large data, volume rendering, ….
  - Interactive Performance (~10Hz is plenty)
- Easy adoptability by actual end users
  - Must be free, have to integrate into commodity vis tools (ParaView, VisIt, VMD, …)
- Easy adoptability for tool developers
  - Must be easy to build, easy to integrate (many platforms, compilers, CPU types, …)
  - Easy to code to, understand, extend, … → API, open source

API & Abstraction Layer

- Same Basic Abstraction Layer as OpenGL (“render frame”)
  - Vis tools/middleware (e.g., VTK) talk to OSPRay through “OSPRay API”
  - Vis tool sets up the scene (geometries, volumes, …) and asks OSPRay to render a frame



API & Abstraction Layer

- Same Basic Abstraction Layer as OpenGL ("render frame")
- OSPRay internally consists of a set of "actors"
  - **Geometries**: Geometric Primitives
    - TriangleMesh, Spheres, Cylinders, StreamLines, Instances, ImplicitIsoSurfaces, …
  - **Volumes**: Scalar Fields that can be sampled (for volume rendering)
    - StructuredVolume variants
    - Prototypically: Unstructured Tet meshes, Chombo Berger/Collela AMR, RBFs, …
  - **Renderers**: Things that trace rays to compute pixel colors
    - "SciVis" (shadows+transparency+phong shading+AO+volume rendering+…)
    - "PathTracer" (guess…)
  - Plus: Cameras, lights, materials, frame buffer, FB pixel-ops, data arrays, …

API & Abstraction Layer

- Same Basic Abstraction Layer as OpenGL ("render frame")
- OSPRay internally consists of a set of "actors"
- API allows to create/parameterize/modify these actors
  - Create actors

    ```
    OSPGeometry spheres = ospNewGeometry("spheres");
    ```
  - Create Data arrays (equivalent of GPGPU "buffers"; often zero-copy)

    ```
    OSPData center = ospNewData(N,OSP_VEC3F,&sphereArray[0]);
    ```
  - Set parameters

    ```
    ospSetData(spheres,"center",center);
    ```
  - "commit" an object (ie, "apply those parameters")

    ```
    ospCommit(spheres);
    ```

API & Abstraction Layer

- Same Basic Abstraction Layer as OpenGL (“render frame”)
- OSPRay internally consists of a set of “actors”
- API allows to create/parameterize/modify these actors
- Once all actors are set:

Render frame…

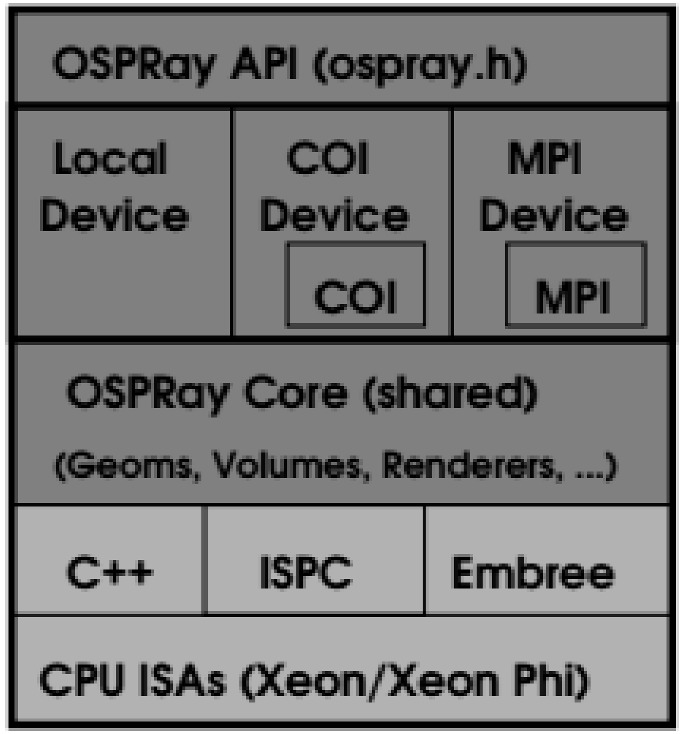
```
ospRenderFrame(fb, renderer, OSP_FB_COLOR);
```

… and map frame buffer

```
void *fb = ospMapFrameBuffer(fb,OSP_FB_COLOR);
glDrawPixels(….)
```

High-Level Architecture

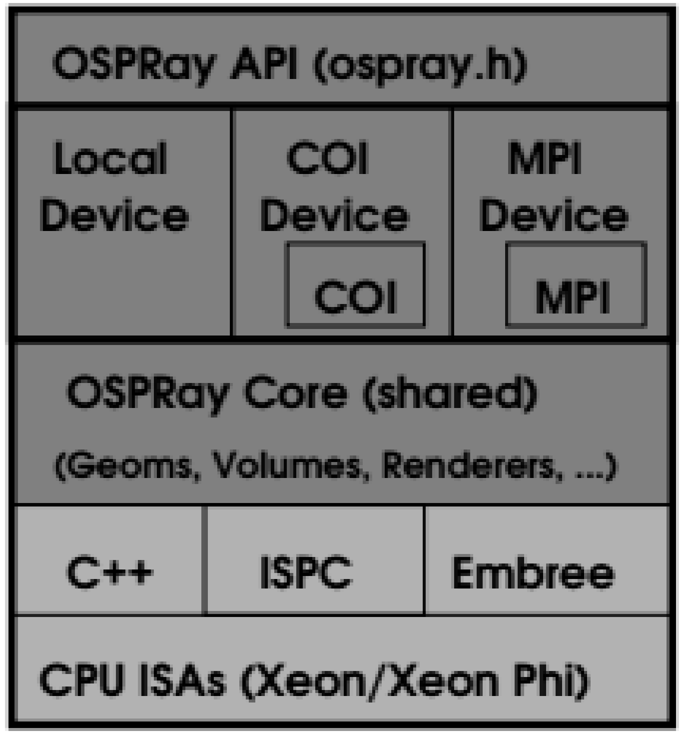
- Internally: Implemented through multiple “devices”
  - Device: abstract object that implements the API calls
    - **Local device** (local node rendering)
    - **Offload device**
    - **MPI device** (MPI-parallel rendering), …
  - Devices build on top of a common core (geometries, volumes, renderers, …)



High-Level Architecture

- Internally: Implemented through multiple “devices”

- Built on top of C++, Embree, and ISPC
  - Embree for accel structure construction and traversal
    - Order 100s of mio’s of prims/sec data struct construction
    - Order 100s of mio’s of rays/sec traversal perf (actual perf depends on model, CPU type, …)
  - ISPC for all performance-critical code (rendering)
  - C++ for high-level system/admin code (MPI communication, …)

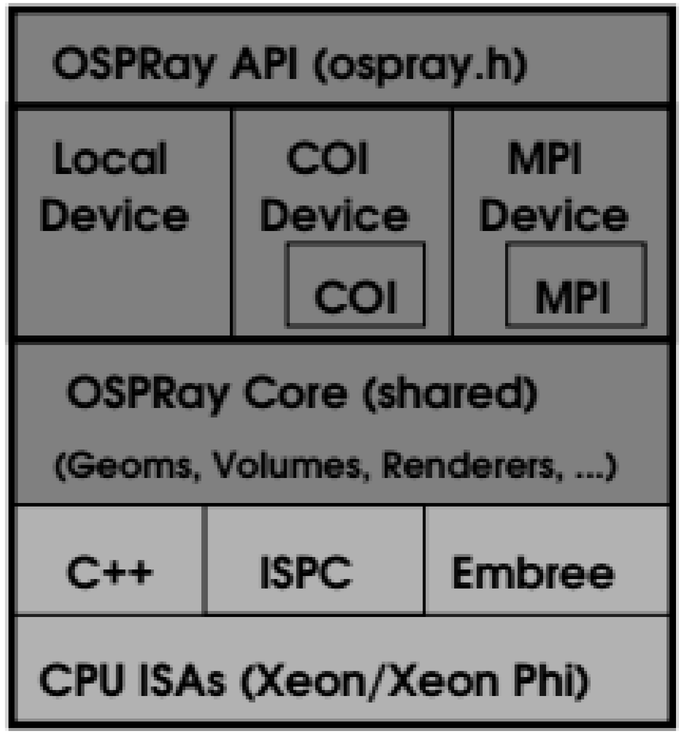


High-Level Architecture

- Internally: Implemented through multiple “devices”
- Built on top of C++, Embree, and ISPC
- Different CPU Archs (Intel Core/Xeon/Xeon Phi) addressed through Embree and ISPC
  - Embree: Hand-coded intrinsics (SSE, AVX, AVX2, AVX-512)
  - ISPC: uses LLVM to emit to SSE, AVX, AVX2, AVX-512
  - All C++ code is completely ISA agnostic (no intrinsics anywhere!)
  - Both ISPC and Embree perform automatic ISA selection

  →OSPRay automatically selects best code based on CPU used

  →Runs on any modern Intel © Xeon© or Intel © Xeon Phi™ system



Current Status

- Version 1.0 released this summer
  - Early prototypes shown since late last year (SC, ISC, …)

Current Status

- Version 1.0 released this summer
- Current version is v1.1.1
  - Lost of bug fixes, optimizations (faster volume rendering: adaptive sampling)
  - Better integration of volume and surface rendering
  - Improvements to build system (binary releases, different platforms, newest embree, etc)
  - Available by default in ParaView 5.2 (click the “render with ospray” box)
  - Integrated (to varying degree) into VMD, VisIt, VL3, EasternGraphics,…

Current Status

- Version 1.0 released this summer
- Current version is v1.1.1
- Supports wide range of platforms
  - OS'es & compilers: Linux, MacOS, and Windows; gcc, clang, msvc, and icc
  - CPUs: Anything SSE4 and newer (in part, including Intel© Xeon Phi™ Knights Landing)

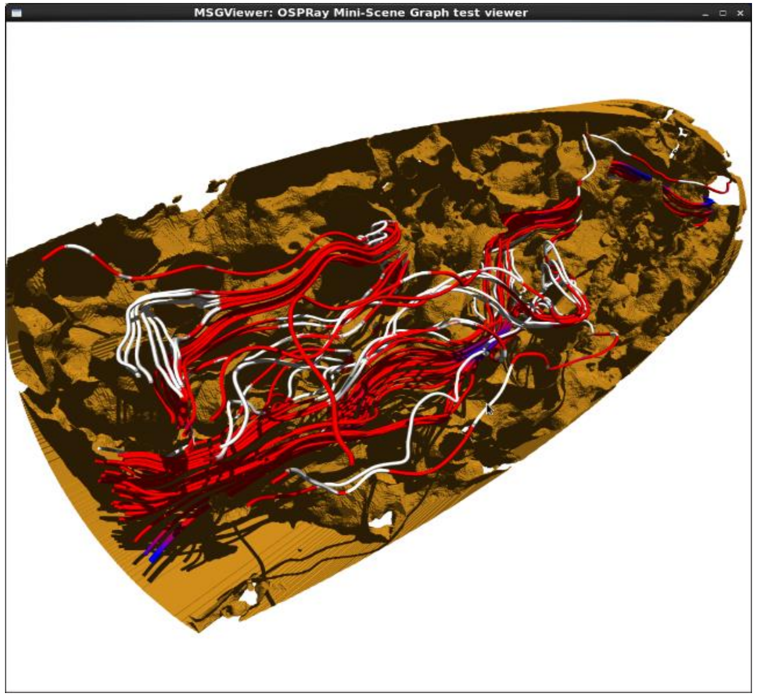
Current Status

- Version 1.0 released this summer
- Current version is v1.1.1
- Supports wide range of platforms
- All fully Open Source (Apache License)
  - Source code hosted on http://ospray.github.io
  - Pre-built binaries - and some demo data sets - on http://www.ospray.org

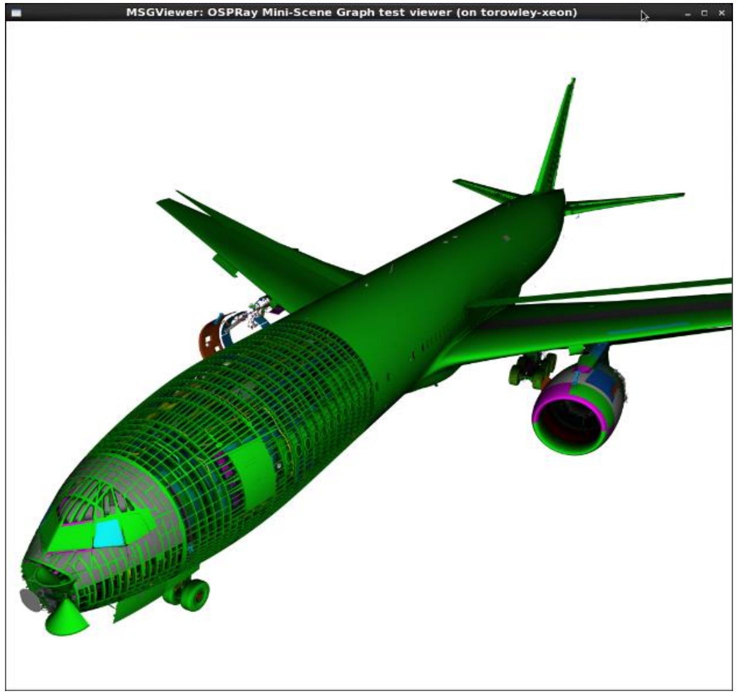
OSPRay Capabilities (v1.1.1)

Capabilities: Large Triangle Meshes

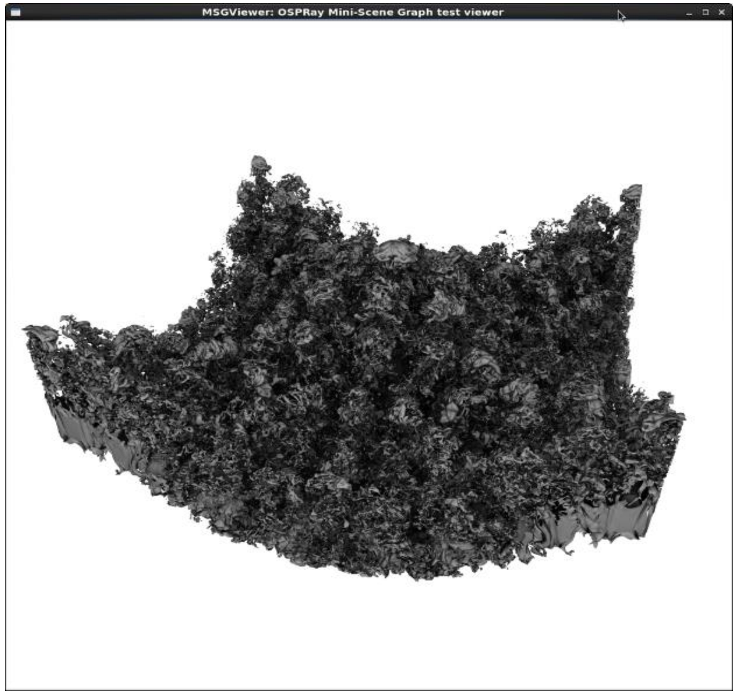
- Pretty much: as much as your main memory can hold (ca 100B/triangle)



“FIU” model (8-80M triangles),
data courtesy Carson Brownlee, TACC, and
Florida International University



Boeing 777 model provided by and used with
permission of Boeing Corp.
340 million triangles, with Ambient Occlusion

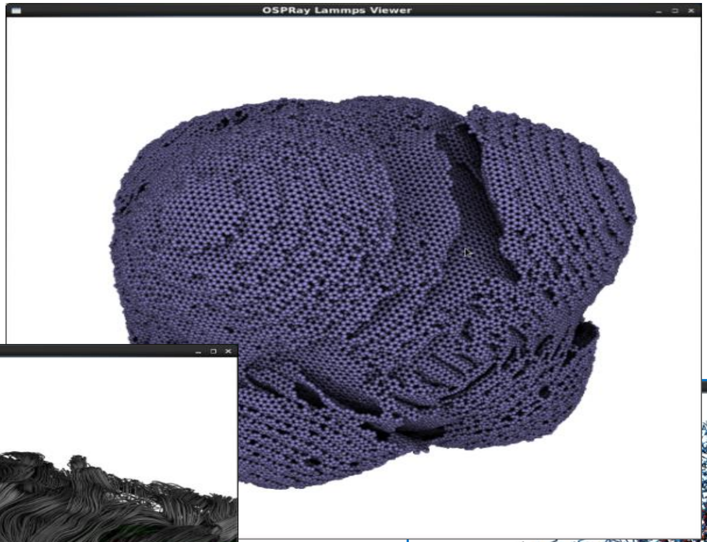


“Richtmeyer-Meshkov Isosurface” model
(ParaView contour from 2k^3 volume)
~290M triangles, with Ambient Occlusion

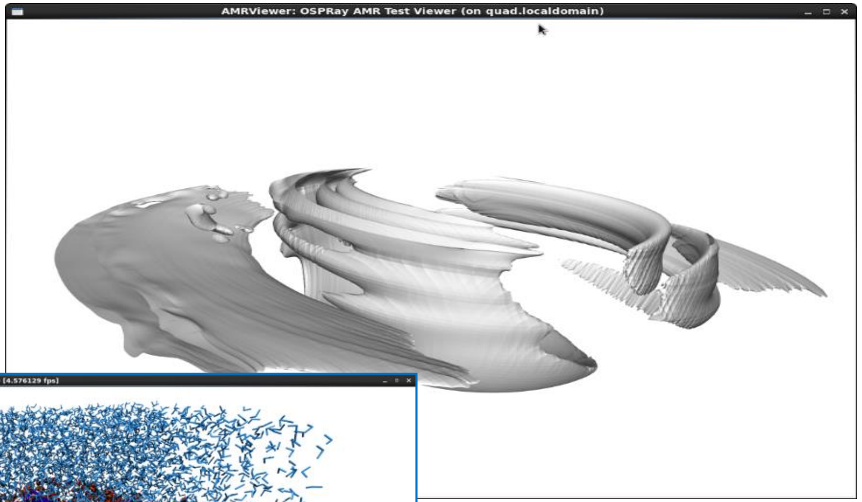
Non-polygonal Geometry…

- Ray tracer can trivially support non-polygonal geometry

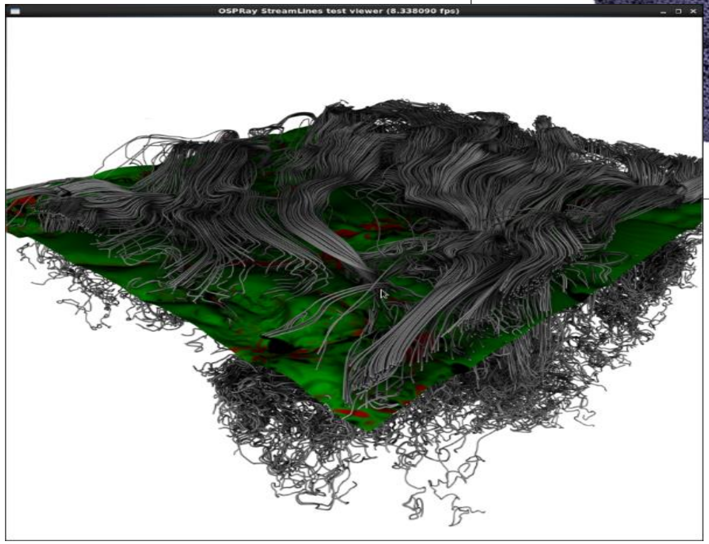
Materials Science: Particles (spheres)
KC Lau, Argonne National Lab)

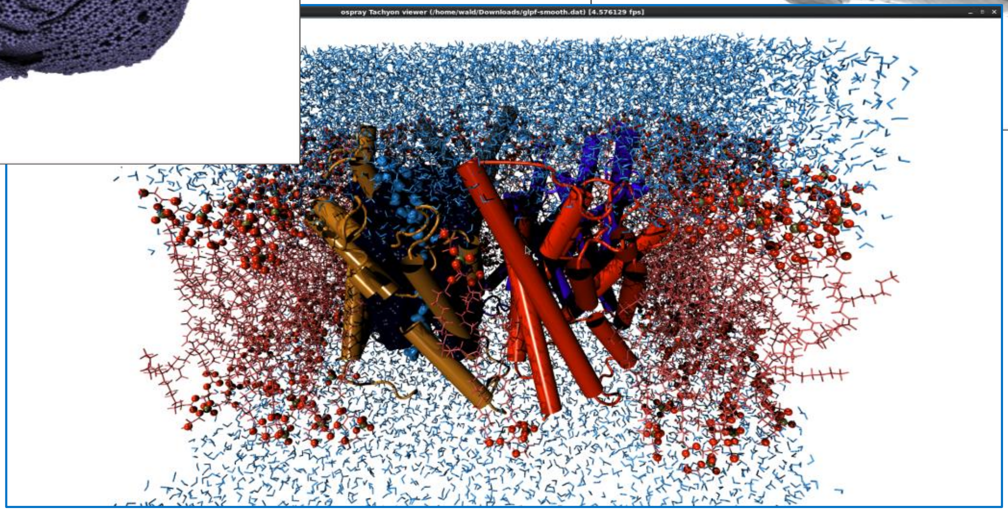


Implicit(!) Iso-Surfaces in Unstructured Volume Dataset
Image courtesy Brad Rathke



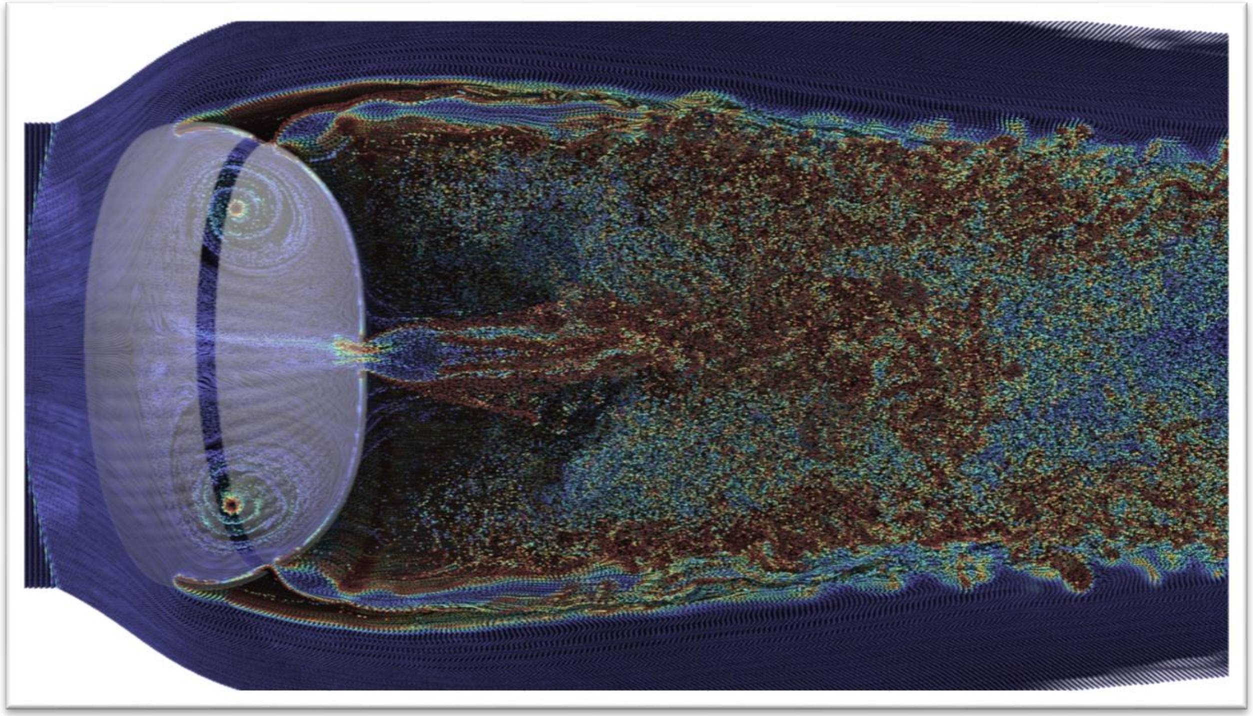
Stream Lines
(Tim Sandstrom, NASA)



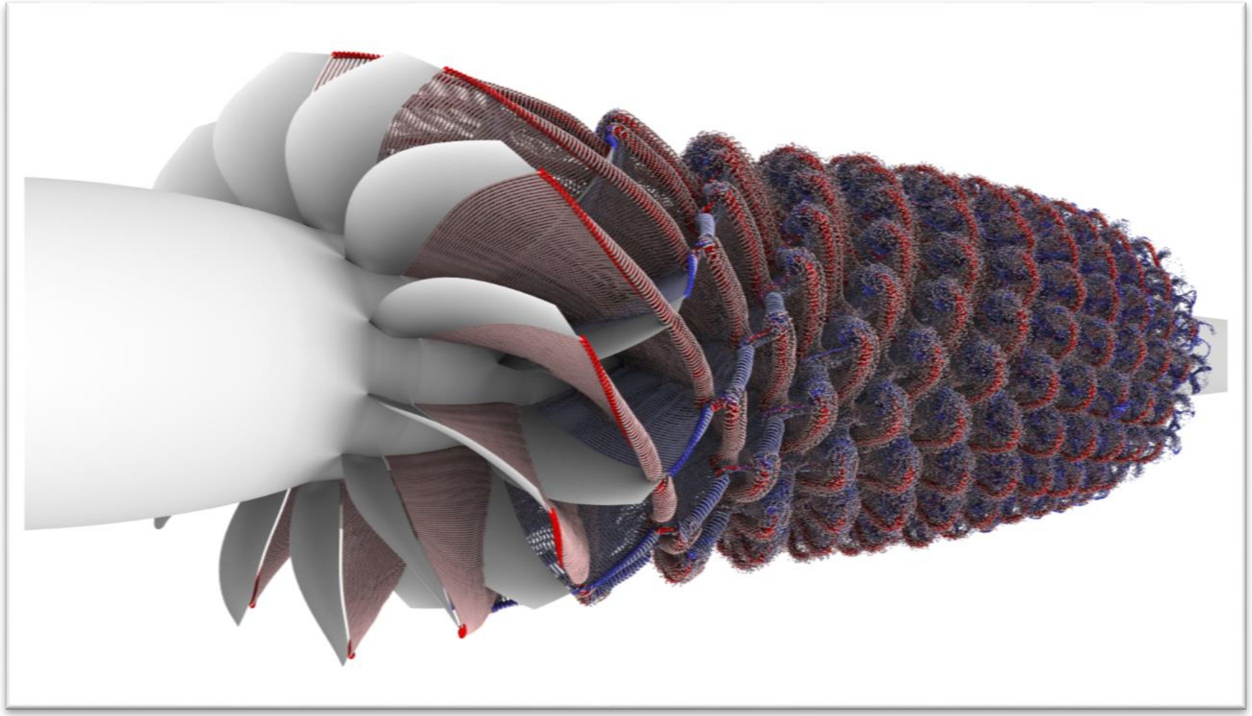


Molecular Modelling:
VMD “glpf” model
Spheres + Cylinders +
Triangles
Model courtesy John Stone

… up to really large particle data



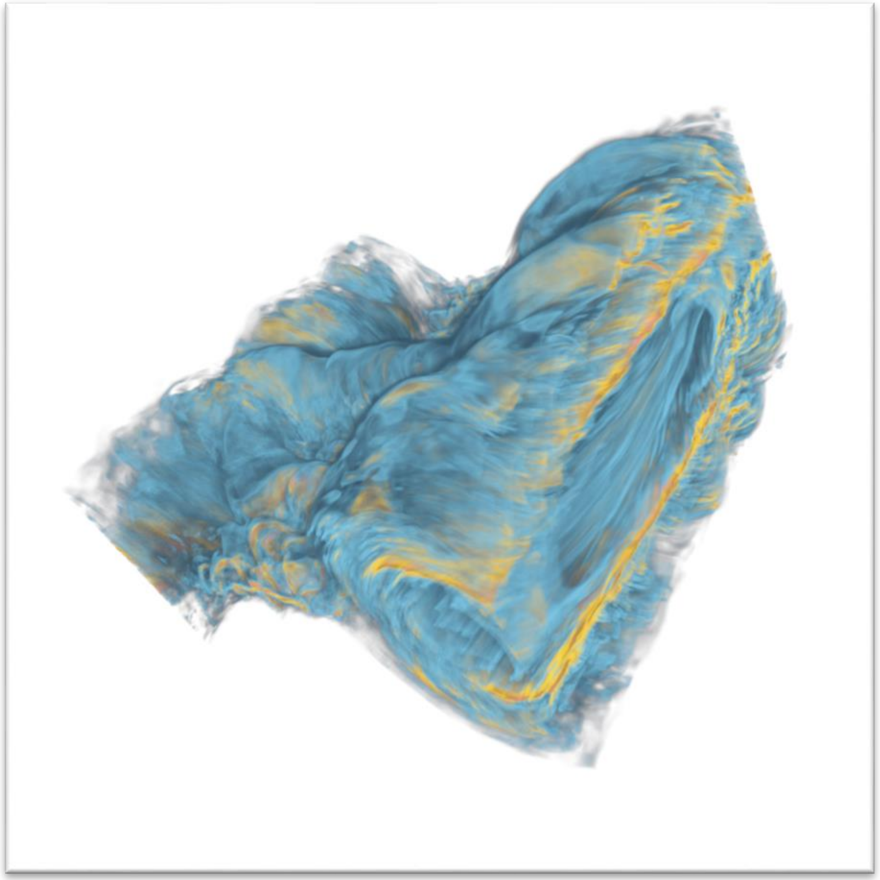
Parachute time series (Tim Sandstrom, NASA Ames)



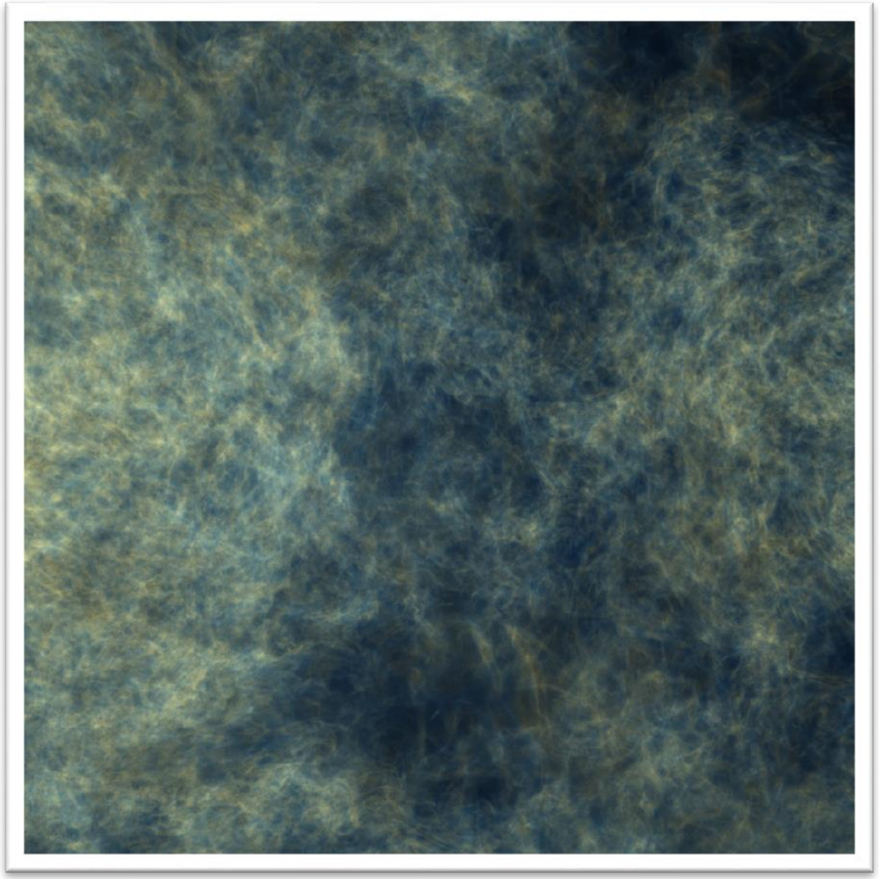
Rotor time series (Tim Sandstrom, NASA Ames)

Volume Rendering

- Again: Pretty much "as big as you can fit into memory"



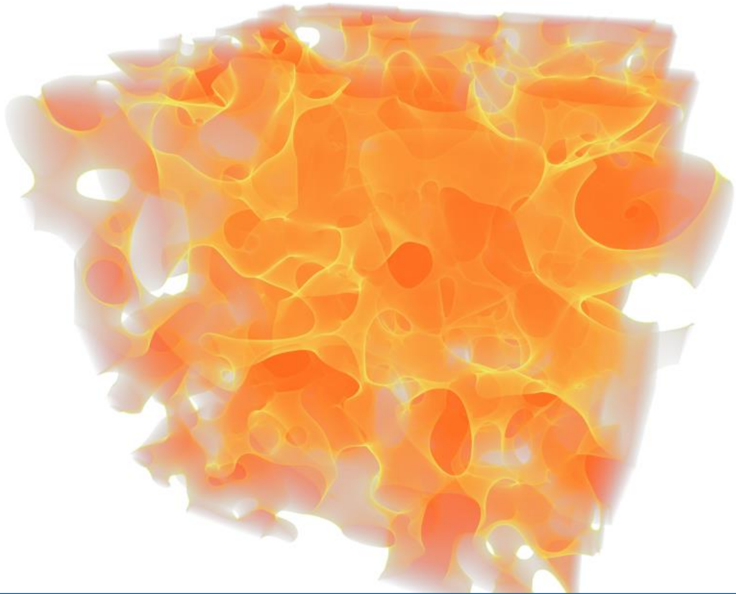
512^3 Magnetic Reconnection Dataset

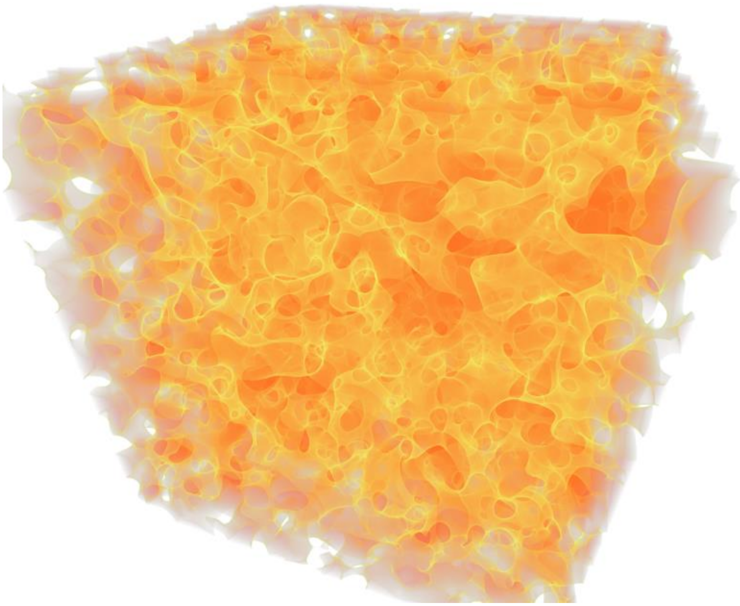


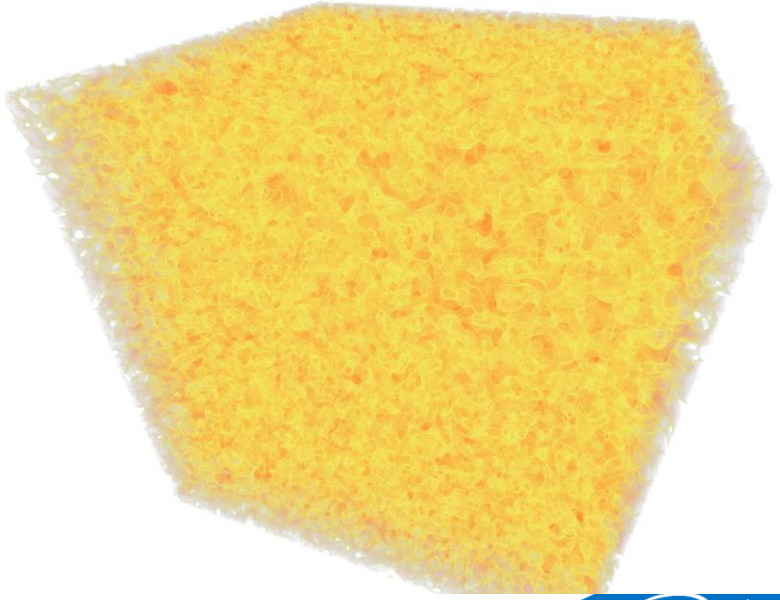
2k^3 Turbulence data

Example: Cosmos "Walls" Demo (SC 15)

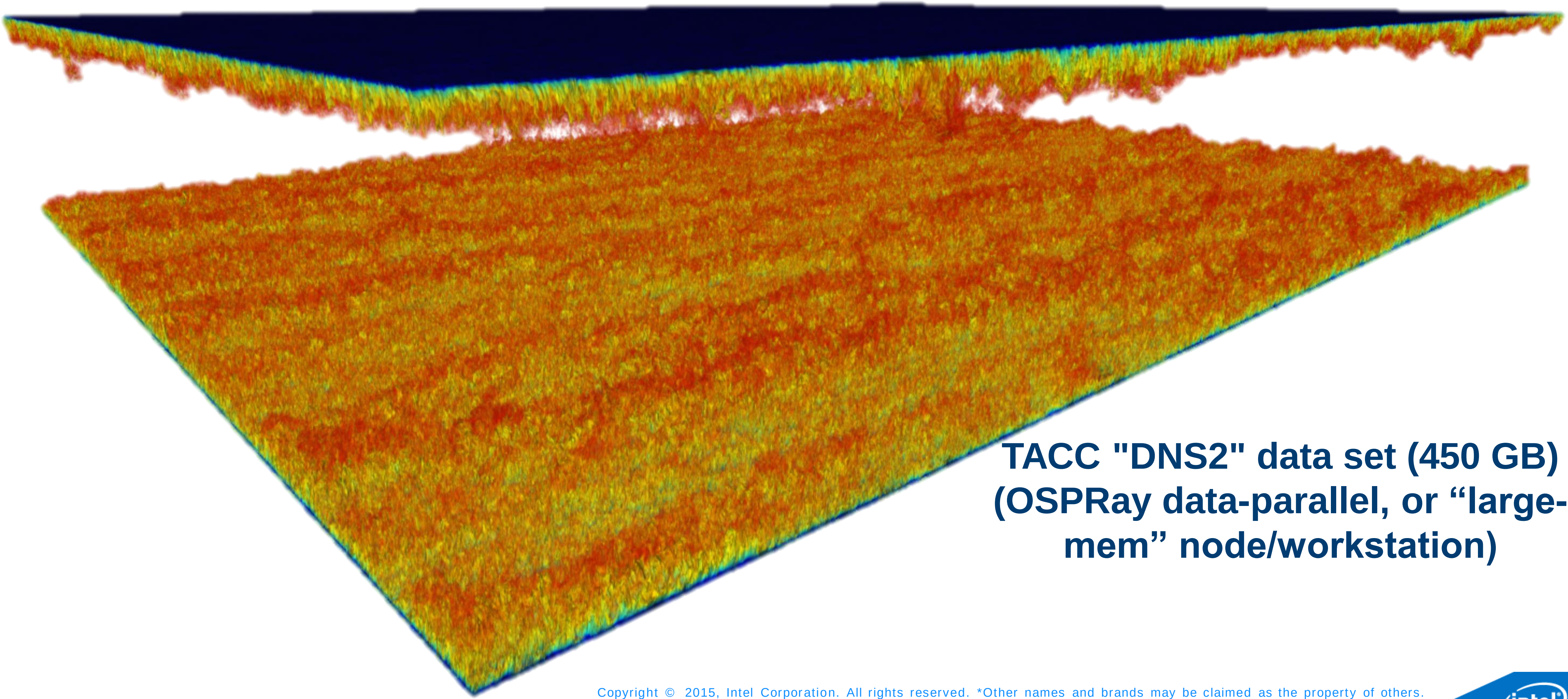
- Large structured volume from UK "Cosmos" team
- Up to 1TB / volume
- Shown on 32TB SGI shared memory machine
- In collaboration w/ Cosmos Team and SGI





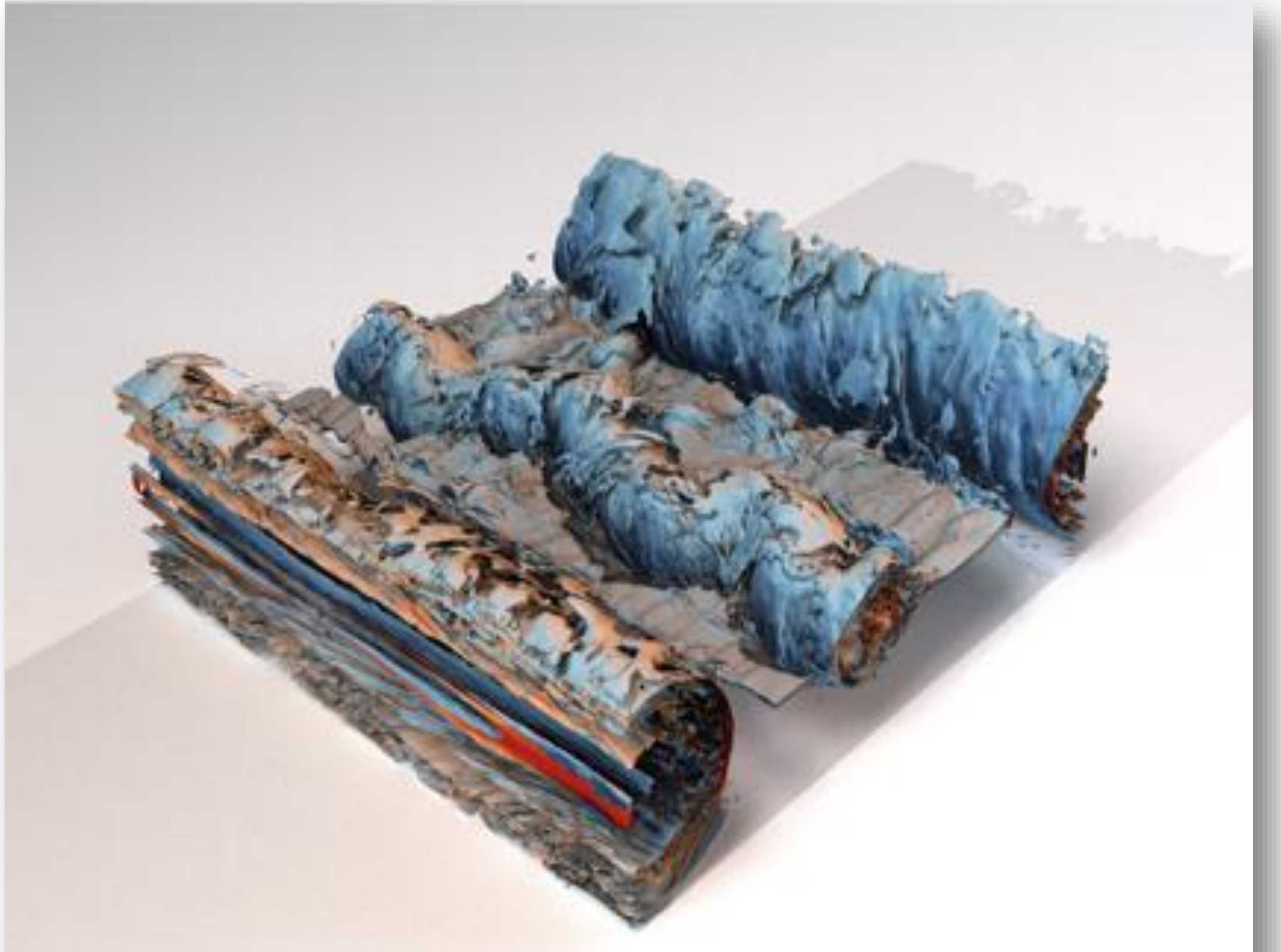


Prototypical MPI-(Data-)Parallel Rendering



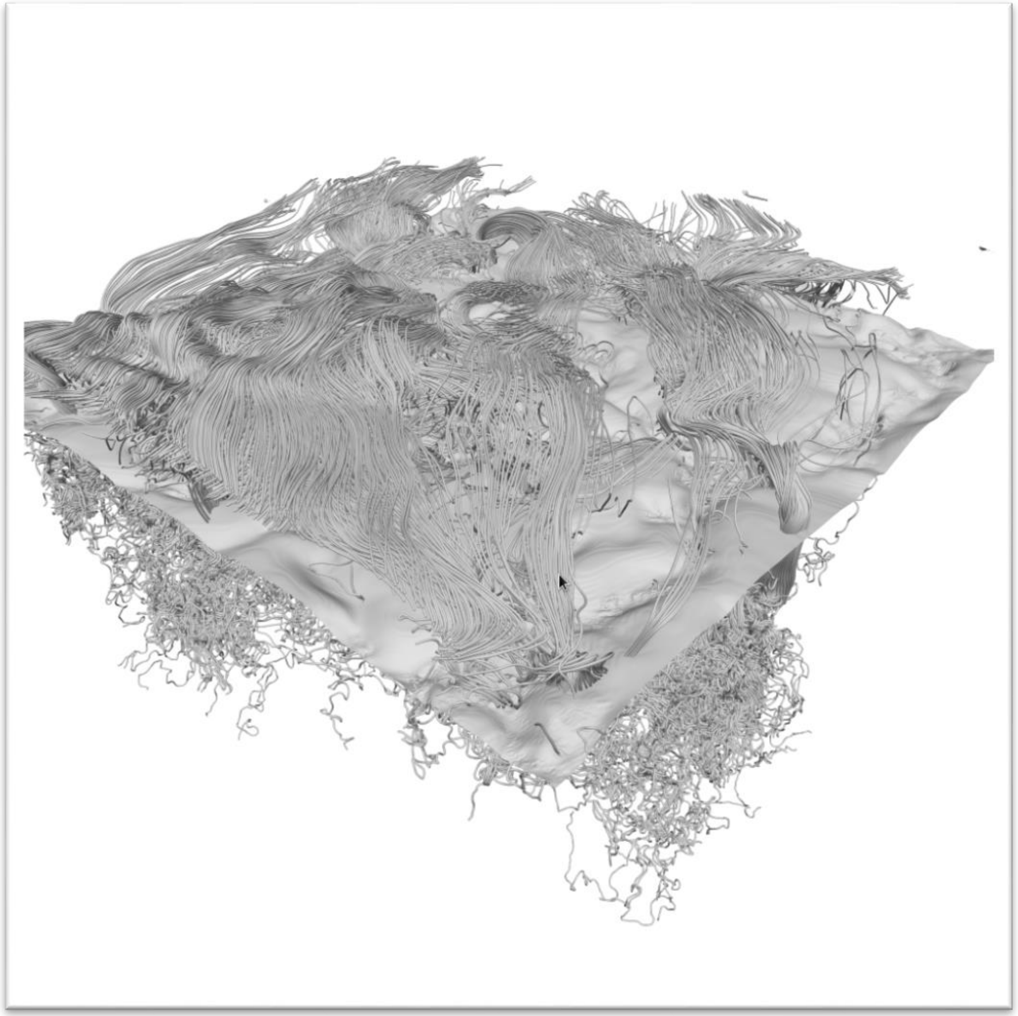
TACC "DNS2" data set (450 GB) (OSPRay data-parallel, or “large-mem” node/workstation)

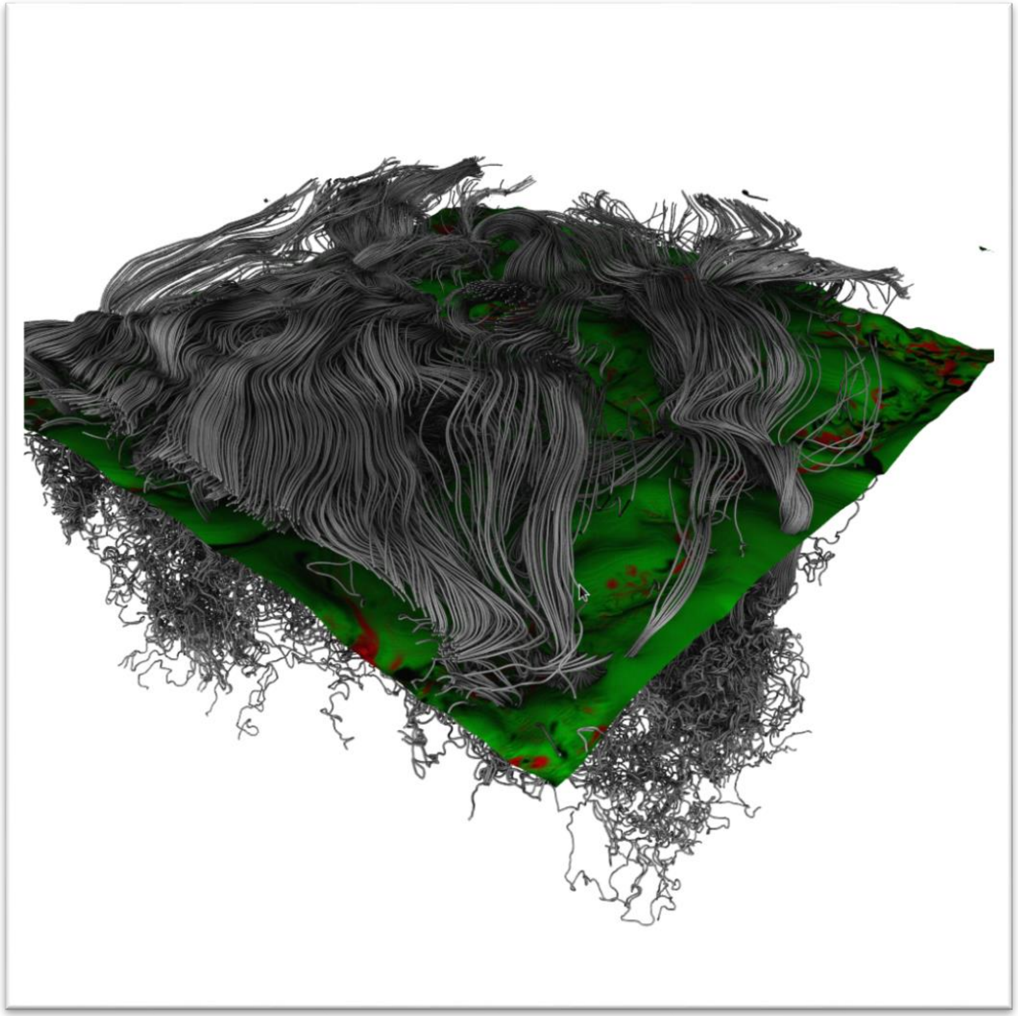
Capabilities: “High-Fidelity” Shading…



Magnetic Reconnection Model, Courtesy Bill Duaghton(LANL) and Berc Geveci(Kitware)

… that really helps in “bringing out the shape” …



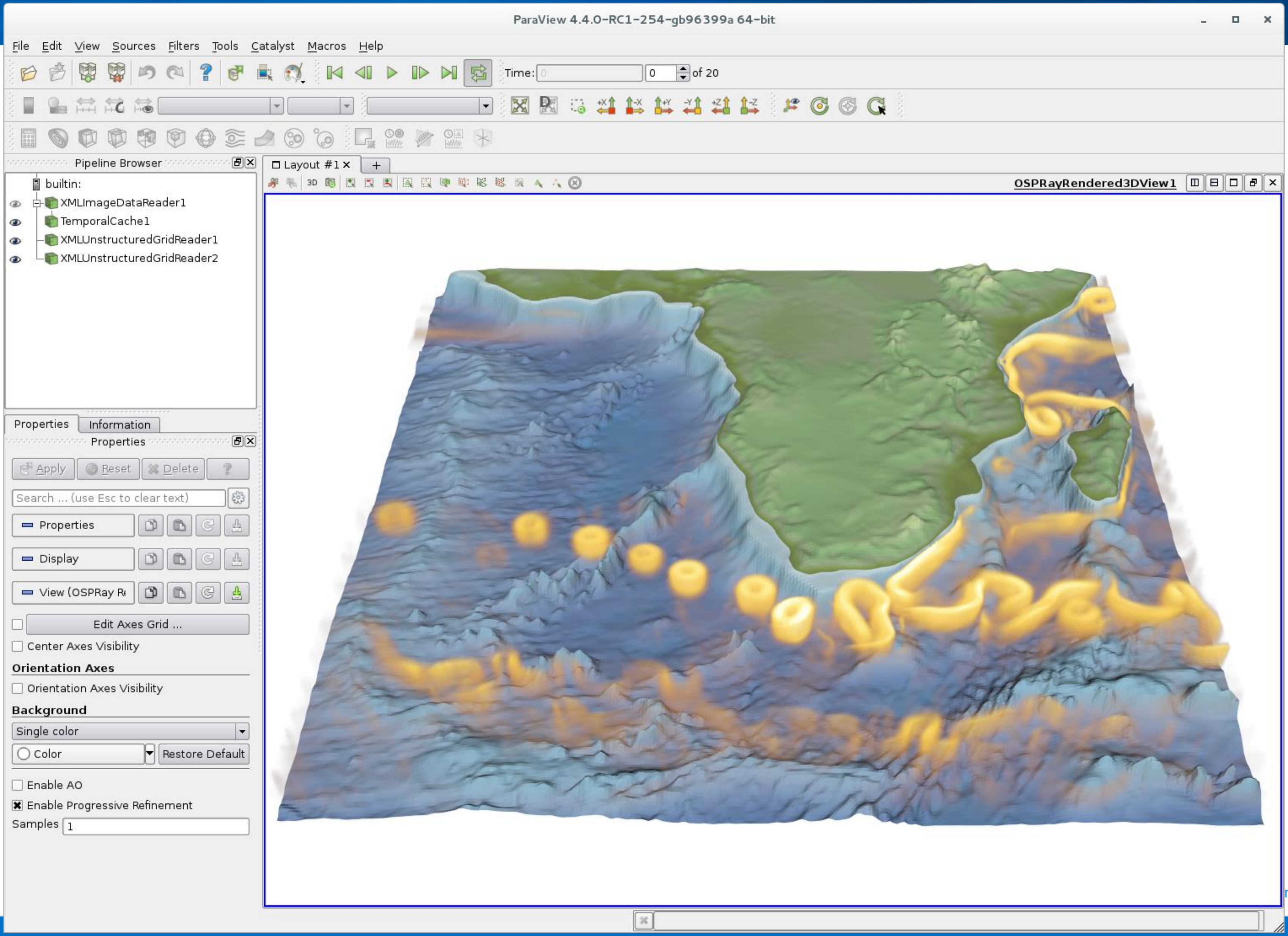


Magnetic field of the Sun (Tim Sandstrom, NASA Ames)

... all the way to photo-realistic path tracer

Model Courtesy EasternGraphics

… and w/ combination of volumes, surfaces, AO, …



Data Courtesy LANL and TACC

Integration into existing vis tools

- ParaView: Ships since 5.1 (surfaces only), latest is ParaView 5.2 (both)
- VisIt: prototypical OSPRay integration exists (→ Hank Childs, Jian Huang)
- VMD: Latest version supports OSPRay renderer
- VTK: Early integration by Dave DeMarle
- More Integrations now in early stages

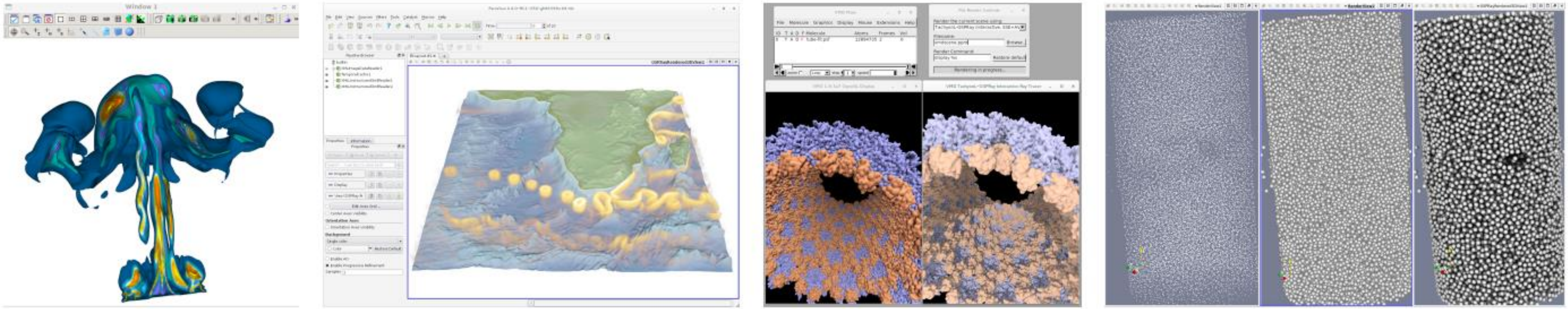


Fig. 8. Though still in Beta release, our OSPRay implementation is already prototypically integrated into three of the most widely used visualization tools, from left to right: VisIt; ParaView; VMD; a prototypical integration into VTK (done by Dave DeMarle at Kitware), showing a simple VTK application using three different VTK renderers—OpenGL points, GL Point Sprites, and OSPRay—side-by-side. Note the improvement in partical locality with ambient occlusion.

Performance

Performance: OSPRay vs Mesa/GPU OpenGL

Methodology: Evaluate for surfaces and volumes, using ParaView

- Pick two representative machines: Workstation & TACC node
  - Both have decent CPUs *and* good GPU
- Pick range of volume data sets
  - From 512MB to 46GB
    (46 GB model gets rendered on 8 nodes in parallel when using GPU; one node w/ OSPRay)
- Pick range of triangle mesh models (contours of volume data)
  - From 20 million tris to > 300 million tris
- Render each w/ ParaView, measure performance
  - OpenGL v1.3 vs 2.0, GPU vs Mesa vs OSPRay

Performance: OSPRay vs Mesa/GPU OpenGL

model	#tris	OpenGL GPU v1.3	OpenGL GPU v2.0	OpenGL Mesa v1.3	OpenGL Mesa v2.0	OSPRay simple	OSPRay AO
High-End Workstation 2×Xeon 2699 v3 "Haswell", 512 GB RAM, NVIDIA Titan X with 12 GB RAM							
isotropic	21.5 M	2.38	83.3	< 1	1.49	47.6	25.6
magnetic	170 M	< 1	10.0	< 1	—*	28.6	20.4
RM	316 M	< 1	4.95	< 1	—*	38.1	20.7
TACC Maverick Node 2×Xeon E5-2680 v2 "IvyBridge", 256 GB RAM, NVIDIA K40m with 12 GB RAM							
isotropic	21.5 M	< 1	25.64	< 1	< 1	19.6	10.4
magnetic	170 M	< 1	8.55	< 1	—*	16.1	11.59
RM	316 M	< 1	3.92	< 1	—*	18.2	8.77

Surface Performance (ParaView)

model	size	OpenGL GPU v1.3	OpenGL GPU v2.0	OpenGL Mesa v1.3	OpenGL Mesa v2.0	OSPRay simple	OSPRay gradient
High-End Workstation 2×Xeon 2699 v3 "Haswell", 512 GB RAM, NVIDIA Titan X with 12 GB RAM							
isotropic	512 MB	30.3	56.5	< 1	2.05	40.98	29.5
magnetic	4 GB	23.5	13.7	< 1	—*	15.2	8.47
RM	8 GB	15.9	2.80	< 1	—*	34.6	25.1
TACC Maverick Node 2×Xeon E5-2680 v2 "IvyBridge", 256 GB RAM, NVIDIA K40m with 12 GB RAM							
isotropic	512 MB	23.3	42.9	< 1	3.66	23.8	14.5
magnetic	4 GB	16.8	6.99	< 1	—*	8.33	4.41
RM	8 GB	7.63	1.73	< 1	—*	12.5	7.35
DNS(sub)	46 GB	9.52†	2.66†	< 1	—*	3.07	1.48

Volume Rendering Performance (ParaView)

- Caveat: Take this with a grain of salt …
  - These are *not* pixel-accurate comparisons (ie, apples vs oranges)
  - Data sets, hardware, etc, may or may not be representative …. "Your mileage may vary"
  - Actual performance by now is outdated, anyway (almost 1 years old by now)

Performance: OSPRay vs Mesa/GPU OpenGL

model	#tris	OpenGL GPU v1.3	OpenGL GPU v2.0	OpenGL Mesa v1.3	OpenGL Mesa v2.0	OSPRay simple	OSPRay AO
High-End Workstation — 2×Xeon 2699 v3 “Haswell”, 512 GB RAM, NVIDIA Titan X with 12 GB RAM							
isotropic	21.5 M	2.38	83.3	<1	1.49	47.6	25.6
magnetic	170 M	<1	10.0	<1	—*	28.6	20.4
RM	316 M	<1	4.95	<1	—*	38.1	20.7
TACC Maverick Node — 2×Xeon E5-2680 v2 “IvyBridge”, 256 GB RAM, NVIDIA K40m with 12 GB RAM							
isotropic	21.5 M	<1	25.64	<1	<1	19.6	10.4
magnetic	170 M	<1	8.55	<1	—*	16.1	11.59
RM	316 M	<1	3.92	<1	—*	18.2	8.77

Surface Performance (ParaView)

model	size	OpenGL GPU v1.3	OpenGL GPU v2.0	OpenGL Mesa v1.3	OpenGL Mesa v2.0	OSPRay simple	OSPRay gradient
High-End Workstation — 2×Xeon 2699 v3 “Haswell”, 512 GB RAM, NVIDIA Titan X with 12 GB RAM							
isotropic	512 MB	30.3	56.5	<1	2.05	40.98	29.5
magnetic	4 GB	23.5	13.7	<1	—*	15.2	8.47
RM	8 GB	15.9	2.80	<1	—*	34.6	25.1
TACC Maverick Node — 2×Xeon E5-2680 v2 “IvyBridge”, 256 GB RAM, NVIDIA K40m with 12 GB RAM							
isotropic	512 MB	23.3	42.9	<1	3.66	23.8	14.5
magnetic	4 GB	16.8	6.99	<1	—*	8.33	4.41
RM	8 GB	7.63	1.73	<1	—*	12.5	7.35
DNS(sub)	46 GB	9.52†	2.66†	<1	—*	3.07	1.48

Volume Rendering Performance (ParaView)

- Caveat: Take this with a grain of salt …
- *But:* we can use this to spot *trends!*

Performance: OSPRay vs Mesa/GPU OpenGL

model	#tris	OpenGL GPU v1.3	OpenGL GPU v2.0	OpenGL Mesa v1.3	OpenGL Mesa v2.0	OSPRay simple	OSPRay AO
High-End Workstation 2×Xeon 2699 v3 "Haswell", 512 GB RAM, NVIDIA Titan X with 12 GB RAM							
isotropic	21.5 M	2.38	83.3	< 1	1.49	47.6	25.6
magnetic	170 M	< 1	10.0	< 1	—*	28.6	20.4
RM	316 M	< 1	4.95	< 1	—*	38.1	20.7
TACC Maverick Node 2×Xeon E5-2680 v2 "IvyBridge", 256 GB RAM, NVIDIA K40m with 12 GB RAM							
isotropic	21.5 M	< 1	25.64	< 1	< 1	19.6	10.4
magnetic	170 M	< 1	8.55	< 1	—*	16.1	11.59
RM	316 M	< 1	3.92	< 1	—*	18.2	8.77

Surface Performance (ParaView)

model	size	OpenGL GPU v1.3	OpenGL GPU v2.0	OpenGL Mesa v1.3	OpenGL Mesa v2.0	OSPRay simple	OSPRay gradient
High-End Workstation 2×Xeon 2699 v3 "Haswell", 512 GB RAM, NVIDIA Titan X with 12 GB RAM							
isotropic	512 MB	30.3	56.5	< 1	2.05	40.98	29.5
magnetic	4 GB	23.5	13.7	< 1	—*	15.2	8.47
RM	8 GB	15.9	2.80	< 1	—*	34.6	25.1
TACC Maverick Node 2×Xeon E5-2680 v2 "IvyBridge", 256 GB RAM, NVIDIA K40m with 12 GB RAM							
isotropic	512 MB	23.3	42.9	< 1	3.66	23.8	14.5
magnetic	4 GB	16.8	6.99	< 1	—*	8.33	4.41
RM	8 GB	7.63	1.73	< 1	—*	12.5	7.35
DNS(sub)	46 GB	9.52†	2.66†	< 1	—*	3.07	1.48

Volume Rendering Performance (ParaView)

- Surfaces: OSPRay greatly outperforms Mesa …

Performance: OSPRay vs Mesa/GPU OpenGL

		OpenGL GPU		OpenGL Mesa		OSPRay	
model	#tris	v1.3	v2.0	v1.3	v2.0	simple	AO
High-End Workstation 2×Xeon 2699 v3 "Haswell", 512 GB RAM, NVIDIA Titan X with 12 GB RAM							
isotropic	21.5 M	2.38	83.3	< 1	1.49	47.6	25.6
magnetic	170 M	< 1	10.0	< 1	—*	28.6	20.4
RM	316 M	< 1	4.95	< 1	—*	38.1	20.7
TACC Maverick Node 2×Xeon E5-2680 v2 "IvyBridge", 256 GB RAM, NVIDIA K40m with 12 GB RAM							
isotropic	21.5 M	< 1	25.64	< 1	< 1	19.6	10.4
magnetic	170 M	< 1	8.55	< 1	—*	16.1	11.59
RM	316 M	< 1	3.92	< 1	—*	18.2	8.77

Surface Performance (ParaView)

		OpenGL GPU		OpenGL Mesa		OSPRay	
model	size	v1.3	v2.0	v1.3	v2.0	simple	gradient
High-End Workstation 2×Xeon 2699 v3 "Haswell", 512 GB RAM, NVIDIA Titan X with 12 GB RAM							
isotropic	512 MB	30.3	56.5	< 1	2.05	40.98	29.5
magnetic	4 GB	23.5	13.7	< 1	—*	15.2	8.47
RM	8 GB	15.9	2.80	< 1	—*	34.6	25.1
TACC Maverick Node 2×Xeon E5-2680 v2 "IvyBridge", 256 GB RAM, NVIDIA K40m with 12 GB RAM							
isotropic	512 MB	23.3	42.9	< 1	3.66	23.8	14.5
magnetic	4 GB	16.8	6.99	< 1	—*	8.33	4.41
RM	8 GB	7.63	1.73	< 1	—*	12.5	7.35
DNS(sub)	46 GB	9.52†	2.66†	< 1	—*	3.07	1.48

Volume Rendering Performance (ParaView)

- Surfaces: OSPRay greatly outperforms Mesa …
  … even with OSPRay doing ambient occlusion!

Performance: OSPRay vs Mesa/GPU OpenGL

model	#tris	OpenGL GPU v1.3	OpenGL GPU v2.0	OpenGL Mesa v1.3	OpenGL Mesa v2.0	OSPRay simple	OSPRay AO
High-End Workstation 2×Xeon 2699 v3 "Haswell", 512 GB RAM, NVIDIA Titan X with 12 GB RAM							
isotropic	21.5 M	2.38	83.3	< 1	1.49	47.6	25.6
magnetic	170 M	< 1	10.0	< 1	—*	28.6	20.4
RM	316 M	< 1	4.95	< 1	—*	38.1	20.7
TACC Maverick Node 2×Xeon E5-2680 v2 "IvyBridge", 256 GB RAM, NVIDIA K40m with 12 GB RAM							
isotropic	21.5 M	< 1	25.64	< 1	< 1	19.6	10.4
magnetic	170 M	< 1	8.55	< 1	—*	16.1	11.59
RM	316 M	< 1	3.92	< 1	—*	18.2	8.77

Surface Performance (ParaView)

model	size	OpenGL GPU v1.3	OpenGL GPU v2.0	OpenGL Mesa v1.3	OpenGL Mesa v2.0	OSPRay simple	OSPRay gradient
High-End Workstation 2×Xeon 2699 v3 "Haswell", 512 GB RAM, NVIDIA Titan X with 12 GB RAM							
isotropic	512 MB	30.3	56.5	< 1	2.05	40.98	29.5
magnetic	4 GB	23.5	13.7	< 1	—*	15.2	8.47
RM	8 GB	15.9	2.80	< 1	—*	34.6	25.1
TACC Maverick Node 2×Xeon E5-2680 v2 "IvyBridge", 256 GB RAM, NVIDIA K40m with 12 GB RAM							
isotropic	512 MB	23.3	42.9	< 1	3.66	23.8	14.5
magnetic	4 GB	16.8	6.99	< 1	—*	8.33	4.41
RM	8 GB	7.63	1.73	< 1	—*	12.5	7.35
DNS(sub)	46 GB	9.52†	2.66†	< 1	—*	3.07	1.48

Volume Rendering Performance (ParaView)

- OSPRay vs GPU-OpenGL: "quite competitive"

Performance: OSPRay vs Mesa/GPU OpenGL

model	#tris	OpenGL GPU v1.3	OpenGL GPU v2.0	OpenGL Mesa v1.3	OpenGL Mesa v2.0	OSPRay simple	OSPRay AO
High-End Workstation 2×Xeon 2699 v3 "Haswell", 512 GB RAM, NVIDIA Titan X with 12 GB RAM							
isotropic	21.5 M	2.38	83.3	< 1	1.49	47.6	25.6
magnetic	170 M	< 1	10.0	< 1	—*	28.6	20.4
RM	316 M	< 1	4.95	< 1	—*	38.1	20.7
TACC Maverick Node 2×Xeon E5-2680 v2 "IvyBridge", 256 GB RAM, NVIDIA K40m with 12 GB RAM							
isotropic	21.5 M	< 1	25.64	< 1	< 1	19.6	10.4
magnetic	170 M	< 1	8.55	< 1	—*	16.1	11.59
RM	316 M	< 1	3.92	< 1	—*	18.2	8.77

Surface Performance (ParaView)

model	size	OpenGL GPU v1.3	OpenGL GPU v2.0	OpenGL Mesa v1.3	OpenGL Mesa v2.0	OSPRay simple	OSPRay gradient
High-End Workstation 2×Xeon 2699 v3 "Haswell", 512 GB RAM, NVIDIA Titan X with 12 GB RAM							
isotropic	512 MB	30.3	56.5	< 1	2.05	40.98	29.5
magnetic	4 GB	23.5	13.7	< 1	—*	15.2	8.47
RM	8 GB	15.9	2.80	< 1	—*	34.6	25.1
TACC Maverick Node 2×Xeon E5-2680 v2 "IvyBridge", 256 GB RAM, NVIDIA K40m with 12 GB RAM							
isotropic	512 MB	23.3	42.9	< 1	3.66	23.8	14.5
magnetic	4 GB	16.8	6.99	< 1	—*	8.33	4.41
RM	8 GB	7.63	1.73	< 1	—*	12.5	7.35
DNS(sub)	46 GB	$9.52^\dagger$	$2.66^\dagger$	< 1	—*	3.07	1.48

Volume Rendering Performance (ParaView)

- OSPRay vs GPU-OpenGL: "quite competitive"
  - Lose "some" for small models …

Performance: OSPRay vs Mesa/GPU OpenGL

model	#tris	OpenGL GPU v1.3	OpenGL GPU v2.0	OpenGL Mesa v1.3	OpenGL Mesa v2.0	OSPRay simple	OSPRay AO
High-End Workstation 2×Xeon 2699 v3 "Haswell", 512 GB RAM, NVIDIA Titan X with 12 GB RAM							
isotropic	21.5 M	2.38	83.3	< 1	1.49	47.6	25.6
magnetic	170 M	< 1	10.0	< 1	—*	28.6	20.4
RM	316 M	< 1	4.95	< 1	—*	38.1	20.7
TACC Maverick Node 2×Xeon E5-2680 v2 "IvyBridge", 256 GB RAM, NVIDIA K40m with 12 GB RAM							
isotropic	21.5 M	< 1	25.64	< 1	< 1	19.6	10.4
magnetic	170 M	< 1	8.55	< 1	—*	16.1	11.59
RM	316 M	< 1	3.92	< 1	—*	18.2	8.77

Surface Performance (ParaView)

model	size	OpenGL GPU v1.3	OpenGL GPU v2.0	OpenGL Mesa v1.3	OpenGL Mesa v2.0	OSPRay simple	OSPRay gradient
High-End Workstation 2×Xeon 2699 v3 "Haswell", 512 GB RAM, NVIDIA Titan X with 12 GB RAM							
isotropic	512 MB	30.3	56.5	< 1	2.05	40.98	29.5
magnetic	4 GB	23.5	13.7	< 1	—*	15.2	8.47
RM	8 GB	15.9	2.80	< 1	—*	34.6	25.1
TACC Maverick Node 2×Xeon E5-2680 v2 "IvyBridge", 256 GB RAM, NVIDIA K40m with 12 GB RAM							
isotropic	512 MB	23.3	42.9	< 1	3.66	23.8	14.5
magnetic	4 GB	16.8	6.99	< 1	—*	8.33	4.41
RM	8 GB	7.63	1.73	< 1	—*	12.5	7.35
DNS(sub)	46 GB	9.52†	2.66†	< 1	—*	3.07	1.48

Volume Rendering Performance (ParaView)

- OSPRay vs GPU-OpenGL: "quite competitive"
  - Lose "some" for small models … but actually win for large(r) ones!

Performance: OSPRay vs Mesa/GPU OpenGL

model	#tris	OpenGL GPU v1.3	OpenGL GPU v2.0	OpenGL Mesa v1.3	OpenGL Mesa v2.0	OSPRay simple	OSPRay AO
High-End Workstation 2×Xeon 2699 v3 "Haswell", 512 GB RAM, NVIDIA Titan X with 12 GB RAM							
isotropic	21.5 M	2.38	83.3	< 1	1.49	47.6	25.6
magnetic	170 M	< 1	10.0	< 1	—*	28.6	20.4
RM	316 M	< 1	4.95	< 1	—*	38.1	20.7
TACC Maverick Node 2×Xeon E5-2680 v2 "IvyBridge", 256 GB RAM, NVIDIA K40m with 12 GB RAM							
isotropic	21.5 M	< 1	25.64	< 1	< 1	19.6	10.4
magnetic	170 M	< 1	8.55	< 1	—*	16.1	11.59
RM	316 M	< 1	3.92	< 1	—*	18.2	8.77

Surface Performance (ParaView)

model	size	OpenGL GPU v1.3	OpenGL GPU v2.0	OpenGL Mesa v1.3	OpenGL Mesa v2.0	OSPRay simple	OSPRay gradient
High-End Workstation 2×Xeon 2699 v3 "Haswell", 512 GB RAM, NVIDIA Titan X with 12 GB RAM							
isotropic	512 MB	30.3	56.5	< 1	2.05	40.98	29.5
magnetic	4 GB	23.5	13.7	< 1	—*	15.2	8.47
RM	8 GB	15.9	2.80	< 1	—*	34.6	25.1
TACC Maverick Node 2×Xeon E5-2680 v2 "IvyBridge", 256 GB RAM, NVIDIA K40m with 12 GB RAM							
isotropic	512 MB	23.3	42.9	< 1	3.66	23.8	14.5
magnetic	4 GB	16.8	6.99	< 1	—*	8.33	4.41
RM	8 GB	7.63	1.73	< 1	—*	12.5	7.35
DNS(sub)	46 GB	9.52†	2.66†	< 1	—*	3.07	1.48

Volume Rendering Performance (ParaView)

- OSPRay vs GPU-OpenGL: "quite competitive"
  - Lose "some" for small models … but actually win for large(r) ones!
  - … until eventually even AO is cheaper than GPU-OpenGL

Performance: OSPRay vs Mesa/GPU OpenGL

model	#tris	OpenGL GPU v1.3	OpenGL GPU v2.0	OpenGL Mesa v1.3	OpenGL Mesa v2.0	OSPRay simple	OSPRay AO
High-End Workstation 2×Xeon 2699 v3 "Haswell", 512 GB RAM, NVIDIA Titan X with 12 GB RAM							
isotropic	21.5 M	2.38	83.3	< 1	1.49	47.6	25.6
magnetic	170 M	< 1	10.0	< 1	—*	28.6	20.4
RM	316 M	< 1	4.95	< 1	—*	38.1	20.7
TACC Maverick Node 2×Xeon E5-2680 v2 "IvyBridge", 256 GB RAM, NVIDIA K40m with 12 GB RAM							
isotropic	21.5 M	< 1	25.64	< 1	< 1	19.6	10.4
magnetic	170 M	< 1	8.55	< 1	—*	16.1	11.59
RM	316 M	< 1	3.92	< 1	—*	18.2	8.77

Surface Performance (ParaView)

model	size	OpenGL GPU v1.3	OpenGL GPU v2.0	OpenGL Mesa v1.3	OpenGL Mesa v2.0	OSPRay simple	OSPRay gradient
High-End Workstation 2×Xeon 2699 v3 "Haswell", 512 GB RAM, NVIDIA Titan X with 12 GB RAM							
isotropic	512 MB	30.3	56.5	< 1	2.05	40.98	29.5
magnetic	4 GB	23.5	13.7	< 1	—*	15.2	8.47
RM	8 GB	15.9	2.80	< 1	—*	34.6	25.1
TACC Maverick Node 2×Xeon E5-2680 v2 "IvyBridge", 256 GB RAM, NVIDIA K40m with 12 GB RAM							
isotropic	512 MB	23.3	42.9	< 1	3.66	23.8	14.5
magnetic	4 GB	16.8	6.99	< 1	—*	8.33	4.41
RM	8 GB	7.63	1.73	< 1	—*	12.5	7.35
DNS(sub)	46 GB	9.52†	2.66†	< 1	—*	3.07	1.48

Volume Rendering Performance (ParaView)

- For Direct Volume Rendering: Pretty much the same story
  - Quite competitive overall
  - Lose some for small models, win for larger ones

Performance: OSPRay vs Mesa/GPU OpenGL

model	#tris	OpenGL GPU v1.3	OpenGL GPU v2.0	OpenGL Mesa v1.3	OpenGL Mesa v2.0	OSPRay simple	OSPRay AO
High-End Workstation 2×Xeon 2699 v3 "Haswell", 512 GB RAM, NVIDIA Titan X with 12 GB RAM							
isotropic	21.5 M	2.38	83.3	< 1	1.49	47.6	25.6
magnetic	170 M	< 1	10.0	< 1	—*	28.6	20.4
RM	316 M	< 1	4.95	< 1	—*	38.1	20.7
TACC Maverick Node 2×Xeon E5-2680 v2 "IvyBridge", 256 GB RAM, NVIDIA K40m with 12 GB RAM							
isotropic	21.5 M	< 1	25.64	< 1	< 1	19.6	10.4
magnetic	170 M	< 1	8.55	< 1	—*	16.1	11.59
RM	316 M	< 1	3.92	< 1	—*	18.2	8.77

Surface Performance (ParaView)

model	size	OpenGL GPU v1.3	OpenGL GPU v2.0	OpenGL Mesa v1.3	OpenGL Mesa v2.0	OSPRay simple	OSPRay gradient
High-End Workstation 2×Xeon 2699 v3 "Haswell", 512 GB RAM, NVIDIA Titan X with 12 GB RAM							
isotropic	512 MB	30.3	56.5	< 1	2.05	40.98	29.5
magnetic	4 GB	23.5	13.7	< 1	—*	15.2	8.47
RM	8 GB	15.9	2.80	< 1	—*	34.6	25.1
TACC Maverick Node 2×Xeon E5-2680 v2 "IvyBridge", 256 GB RAM, NVIDIA K40m with 12 GB RAM							
isotropic	512 MB	23.3	42.9	< 1	3.66	23.8	14.5
magnetic	4 GB	16.8	6.99	< 1	—*	8.33	4.41
RM	8 GB	7.63	1.73	< 1	—*	12.5	7.35
DNS(sub)	46 GB	9.52†	2.66†	< 1	—*	3.07	1.48

Volume Rendering Performance (ParaView)

- Interesting when volume no longer fits into GPU memory:
  - Single-node OSPRay "competitive with" eight(!)-node parallel GPU rendering

Summary

Summary

- OSPRay: A CPU Ray Tracing Engine for Sci-Vis Rendering
  - → Enable efficient rendering on CPUs
  - → Bring Ray Tracing / High Fidelity Rendering to Scientific Visualization

Summary

- OSPRay: A CPU Ray Tracing Engine for Sci-Vis Rendering
  - Runs on pretty much any modern CPU, O/S, compiler, …
  - Integrated into ParaView, VisIt, VMD, VTK, …
  - All free, all available in open source (http://ospray.github.io)

Summary

- OSPRay: A CPU Ray Tracing Engine for Sci-Vis Rendering

- Performance:
  - quite competitive even if GPU *is* available…
  - … and particularly useful when for HPC nodes that do *not* have any

    → Turns GPU-less compute node into capable rendering node

    (Plus: more options for higher fidelity, other prim types, larger models, …)

Finally: Request for feed-back

If you find this interesting, then please:

- Try it out (→ http://www.ospray.org)

- If something doesn't work, file it on github!
  (if we don't know it's broken, we cannot fix it!)
- If something is missing, let us know
  (if we don't know it's missing …)
- And: Tell us what you did/do with it!

Questions?

Legal Notices and Disclaimers

- Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. No computer system can be absolutely secure. Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors.

- Performance tests, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Legal Disclaimer and Optimization Notice

INFORMATION IN THIS DOCUMENT IS PROVIDED "AS IS". NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804