Introduction to Scientific Visualization

Kelly Gaither

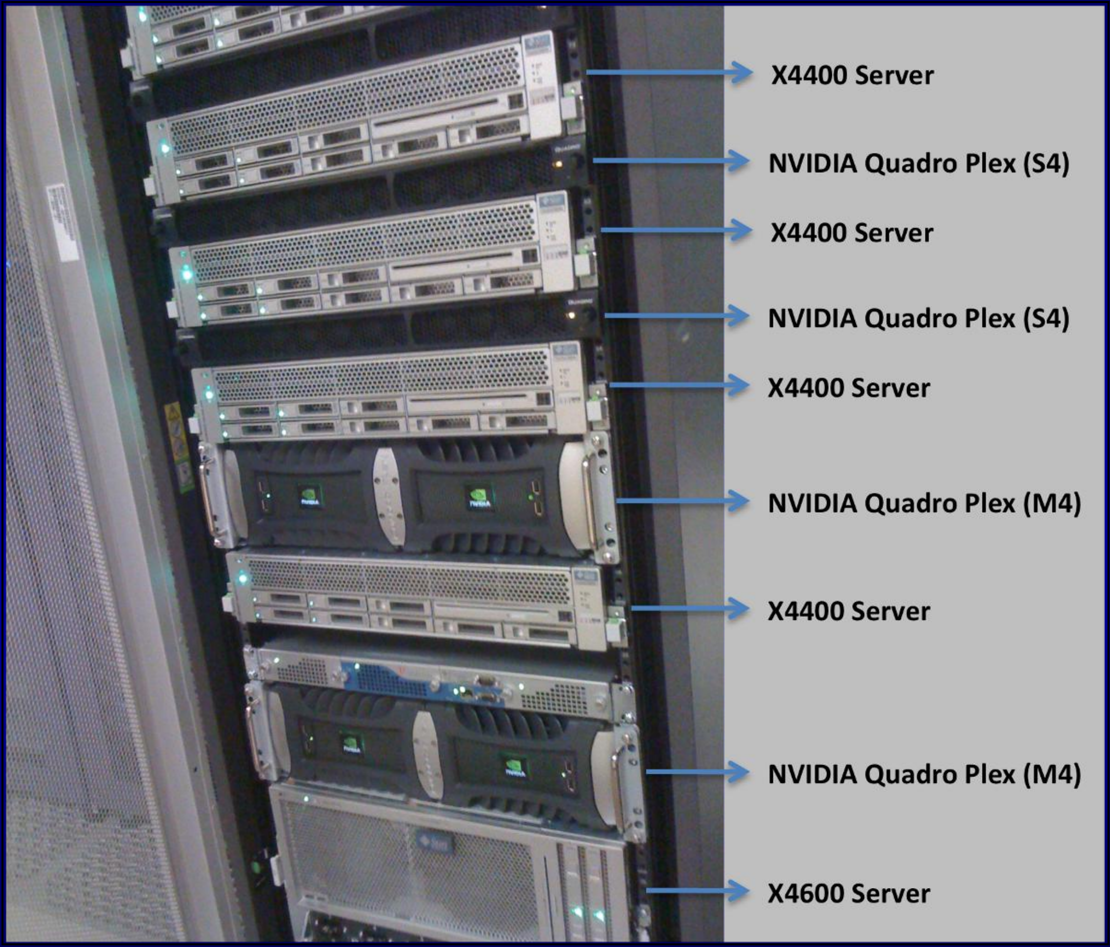
September 2, 2010

Longhorn Visualization and Data Analysis

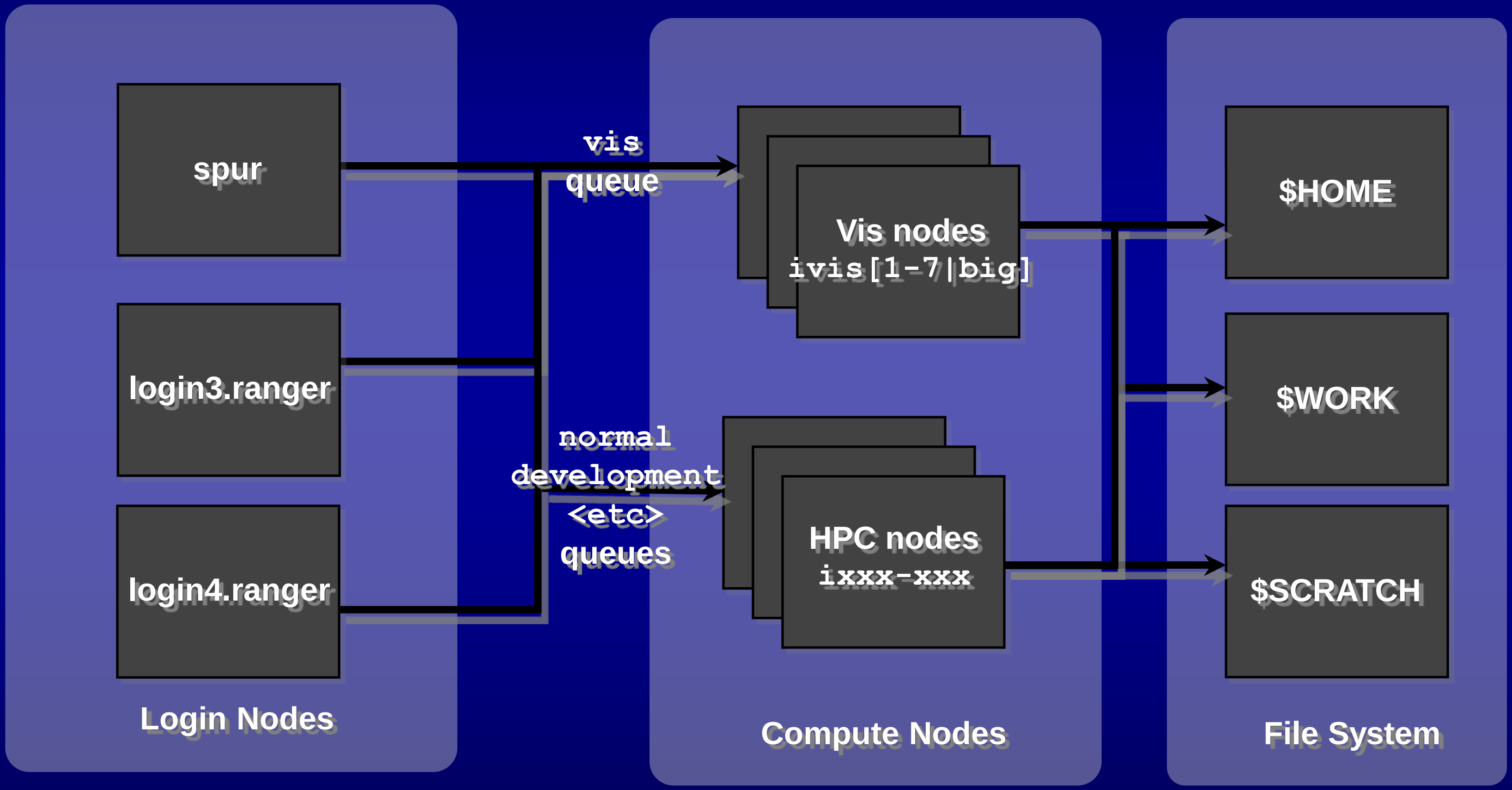
- In November 2008, NSF accepted proposals for the Extreme Digital Resources for Science and Engineering
- The Longhorn project was proposed as a next generation response to TeraGrid’s growing visualization and data analysis needs

Spur - Visualization System

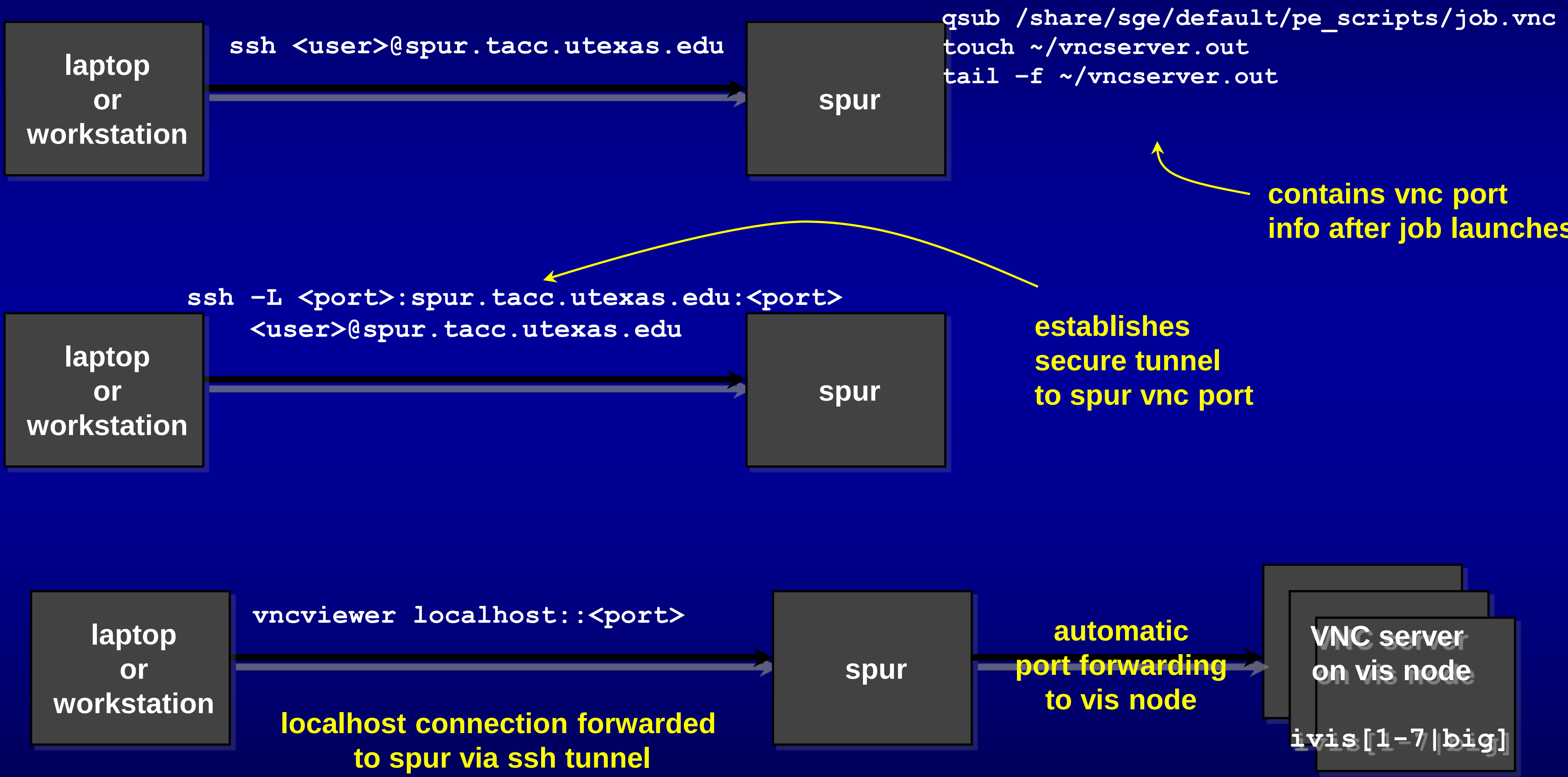
- 128 cores, 1 TB distributed memory, 32 GPUs
- `spur.tacc.utexas.edu`
  login node, no GPUs
  don't run apps here!
- `ivisbig.ranger`
  Sun Fire X4600 server
  - 8 AMD Opteron dual-core CPUs @ 3 GHz
  - 256 GB memory
  - 4 NVIDIA FX5600 GPUs
- `ivis[1-7].ranger`
  Sun Fire X4440 server
  - 4 AMD Opteron quad-core CPUs @ 2.3 GHz
  - 128 GB memory
  - 4 NVIDIA FX5600 GPUs



Spur / Ranger topology



Connecting to Spur



```
ssh <user>@spur.tacc.utexas.edu
```

```
qsub /share/sge/default/pe_scripts/job.vnc
touch ~/vncserver.out
tail -f ~/vncserver.out
```

contains vnc port
info after job launche

```
ssh -L <port>:spur.tacc.utexas.edu:<port>
   <user>@spur.tacc.utexas.edu
```

establishes
secure tunnel
to spur vnc port

```
vncviewer localhost::<port>
```

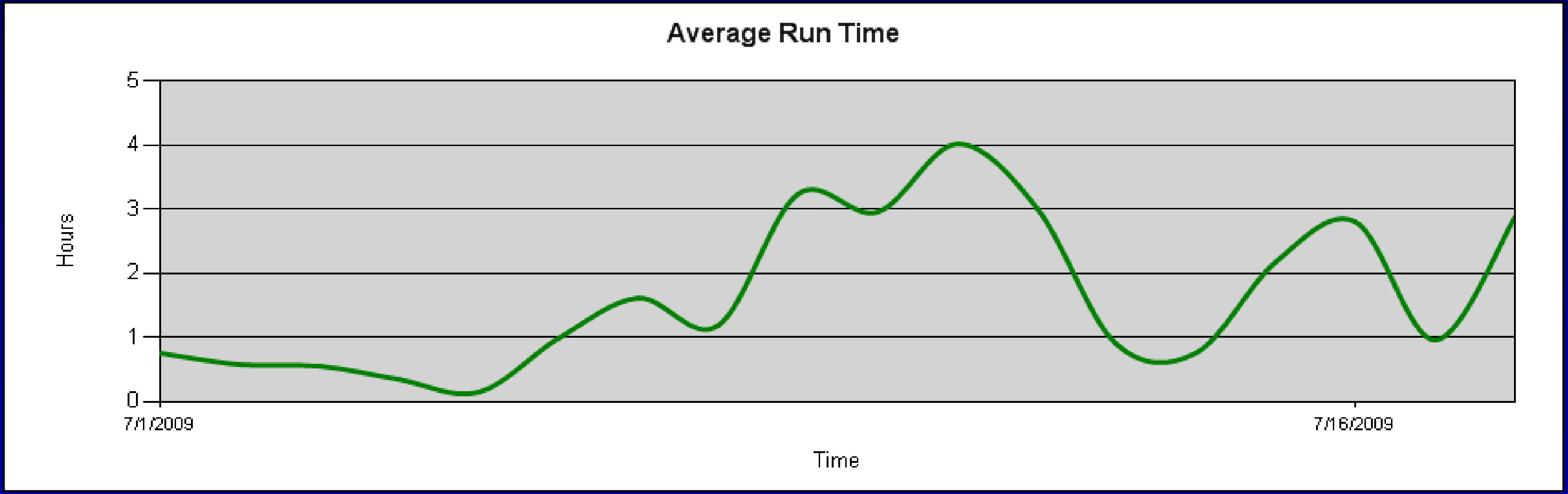
localhost connection forwarded
to spur via ssh tunnel

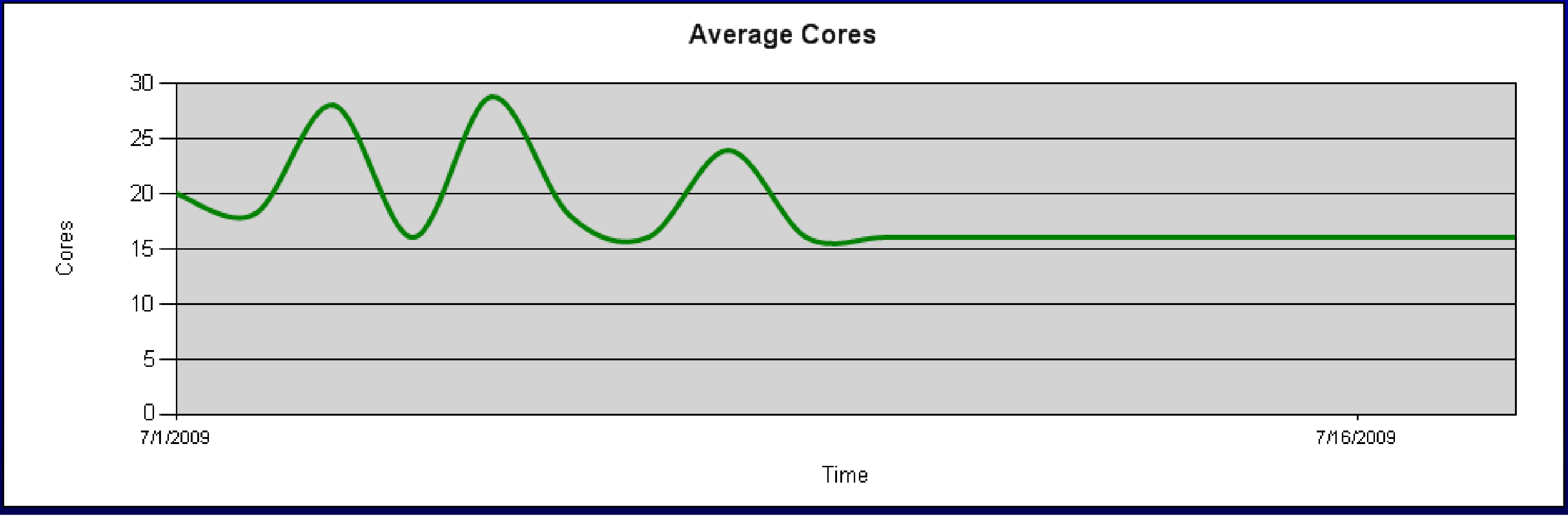
automatic
port forwarding
to vis node

VNC server
on vis node

`ivis[1-7|big]`

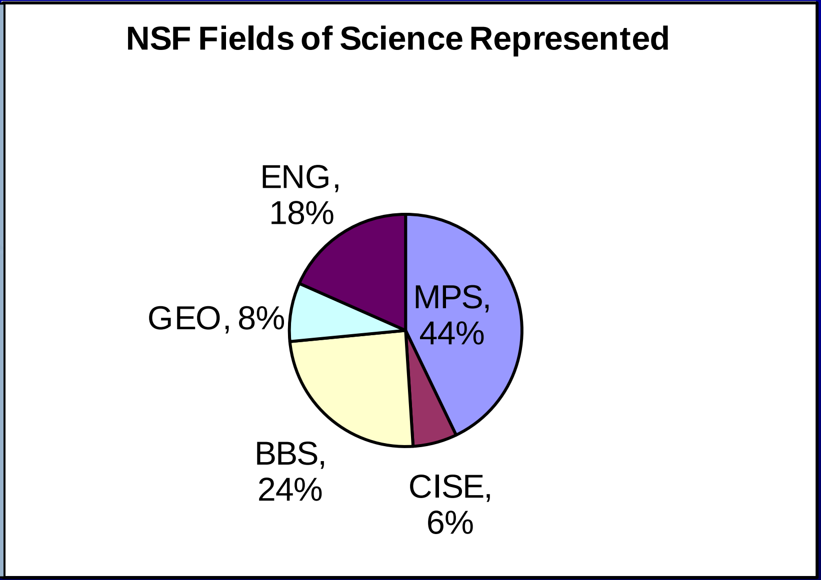
Spur Usage


Average Run Time
Hours
5
4
3
2
1
0
7/1/2009
7/16/2009
Time


Average Cores
Cores
30
25
20
15
10
5
0
7/1/2009
7/16/2009
Time

XD Vis Requirements Analysis

- Surveyed members of the science community via personal interviews and email surveys
- Received ~60 individual responses



XD Vis Requirements Analysis

Requirement	**% Users Requested**
User Support and Consulting	96%
Large-Scale DAV Tools/Resources	39%
Remote/Collaborative DAV Services	27%
Computational Steering	10%
In-simulation DAV Tools	6%
Tools for 3D Measurement and Query	6%
Tools for Multiple Length and Time Scales	6%

(DAV = Data Analysis/Visualization)

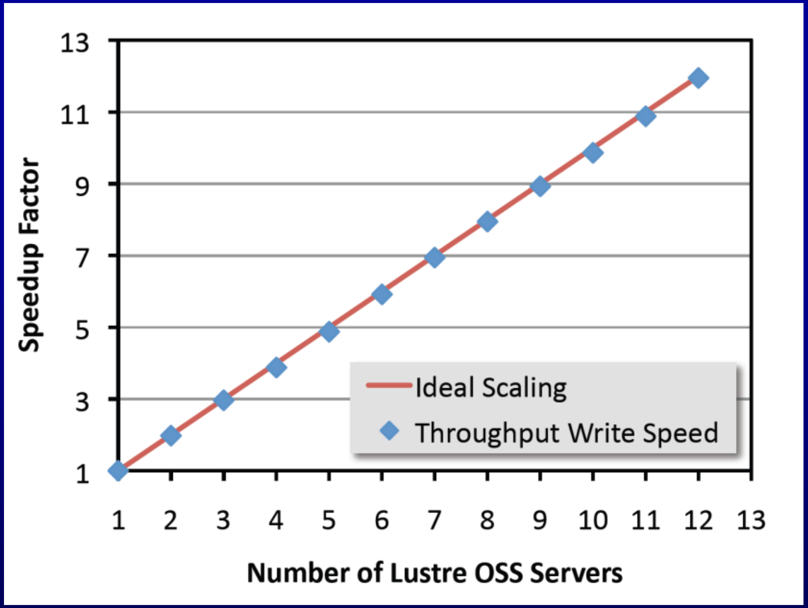
Longhorn Configuration

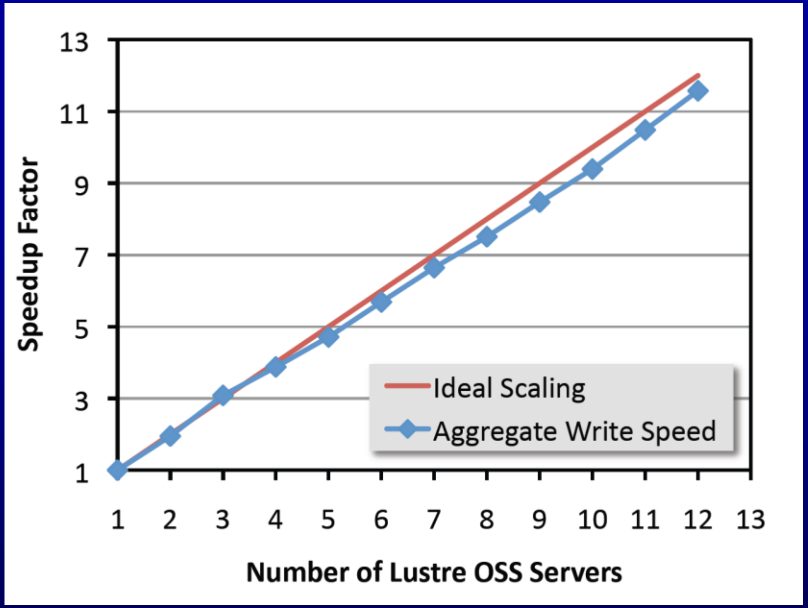
(256 Nodes, 2048 Cores, 512 GPUs, 14.5 TG Aggregate Memory)

- 256 Dell Quad Core Intel Nehalem Nodes
  - 240 Nodes
    - Dual socket, quad core per socket: 8 cores/node
    - 48 GB shared memory/node (6 GB/core)
    - 73 GB Local Disk
    - 2 Nvidia GPUs/node (FX 5800 - 4GB RAM)
  - 16 Nodes
    - Dual socket, quad core per socket: 8 cores/node
    - 144 GB shared memory/node (18 GB/core)
    - 73 GB Local Disk
    - 2 Nvidia GPUs/node (FX 5800 – 4GB RAM)
  - ~14.5 TB aggregate memory
- QDR InfiniBand Interconnect
- Direct Connection to Ranger's Lustre Parallel File System
- 10G Connection to 210 TB Local Lustre Parallel File System
- Jobs launched through SGE

Longhorn’s Lustre File System ($SCRATCH)

- OSS’s on Longhorn are built on Dell Nehalem Servers Connected to MD10000 Storage Vaults
- 15 Drives Total Configured into 2 Raid5 pairs with a Wandering Spare
- Peak Throughput Speed of the File System is 5.86 GB/sec
- Peak Aggregate Speed of the File System is 5.43 GB/sec





Longhorn Partners and Roles:

- TACC (Kelly Gaither – PI)
  - **Longhorn** machine deployment
  - User support
  - Visualization and Data Analysis portal development
  - Software/Tool development
- NCAR (John Clyne – CoPI)
  - User support
  - VAPOR Enhancements
- University of Utah (Valerio Pascucci – CoPI, Chuck Hansen)
  - User support
  - Software Integration of RTRT and topological analysis

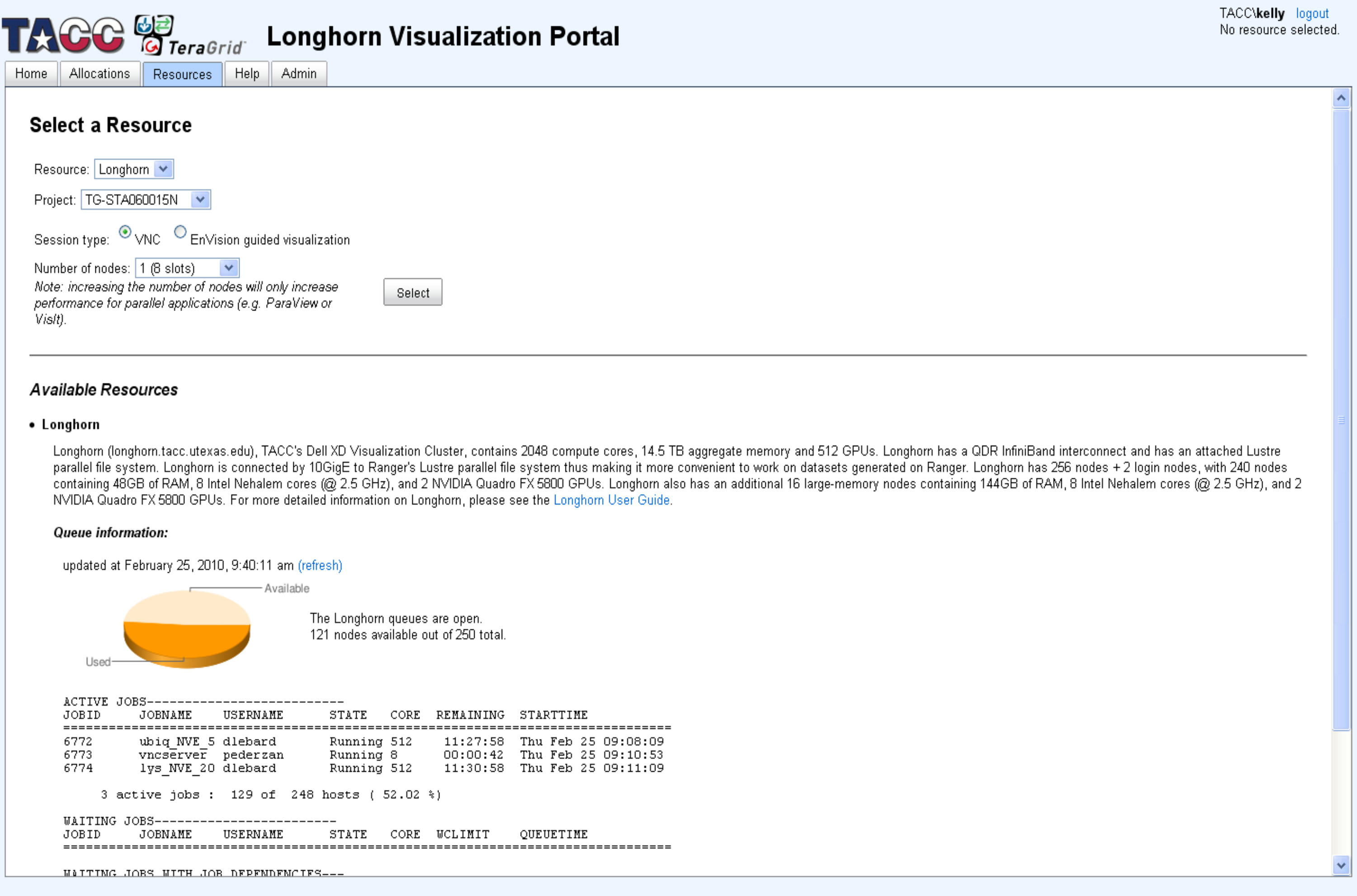
Longhorn Partners and Roles:

- Purdue University (David Ebert – CoPI)
  - User support
  - Integration of visual analytics software
- UC Davis (Hank Childs – Chief Software Integration Architect)
  - Directly facilitate tools being integrated into the VisIt software suite
- SURA (Linda Akli – MSI Outreach/Broadening Participation)

Longhorn Usage Modalities:

- Remote/Interactive Visualization
  - Highest priority jobs
  - Remote/Interactive capabilities facilitated through VNC
  - Run on 4 hour time limit
- GPGPU jobs
  - Run on a lower priority than the remote/interactive jobs
  - Run on 12 hour time limit
- CPU jobs with higher memory requirements
  - Run on lowest priority when neither remote/interactive nor GPGPU jobs are waiting in the queue
  - Run 12 hour time limit

Longhorn User Portal



Longhorn Visualization Portal

TACC**kelly** logout
No resource selected.

Select a Resource

Resource: Longhorn

Project: TG-STA060015N

Session type: ● VNC ○ EnVision guided visualization

Number of nodes: 1 (8 slots)

Note: increasing the number of nodes will only increase performance for parallel applications (e.g. ParaView or VisIt).

Available Resources

- **Longhorn**

  Longhorn (longhorn.tacc.utexas.edu), TACC's Dell XD Visualization Cluster, contains 2048 compute cores, 14.5 TB aggregate memory and 512 GPUs. Longhorn has a QDR InfiniBand interconnect and has an attached Lustre parallel file system. Longhorn is connected by 10GigE to Ranger's Lustre parallel file system thus making it more convenient to work on datasets generated on Ranger. Longhorn has 256 nodes + 2 login nodes, with 240 nodes containing 48GB of RAM, 8 Intel Nehalem cores (@ 2.5 GHz), and 2 NVIDIA Quadro FX 5800 GPUs. Longhorn also has an additional 16 large-memory nodes containing 144GB of RAM, 8 Intel Nehalem cores (@ 2.5 GHz), and 2 NVIDIA Quadro FX 5800 GPUs. For more detailed information on Longhorn, please see the Longhorn User Guide.

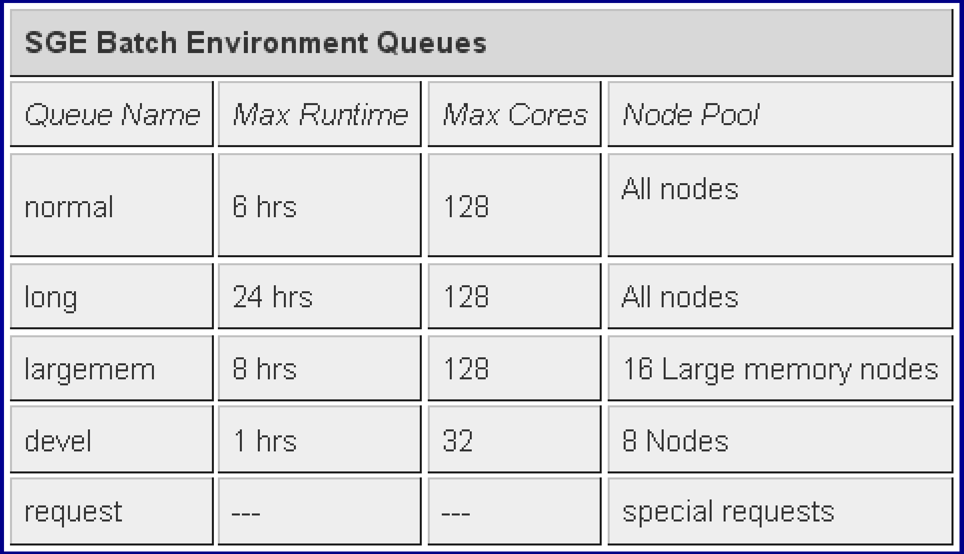
  Queue information:

  updated at February 25, 2010, 9:40:11 am (refresh)

  The Longhorn queues are open.
  121 nodes available out of 250 total.

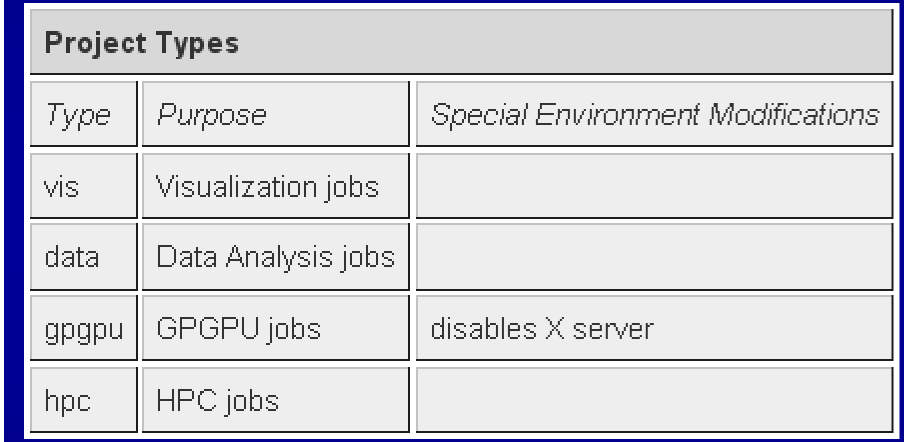
```
ACTIVE JOBS--------------------------
JOBID     JOBNAME    USERNAME      STATE   CORE  REMAINING  STARTTIME
================================================================================
6772      ubiq_NVE_5 dlebard       Running 512    11:27:58  Thu Feb 25 09:08:09
6773      vncserver  pederzan      Running 8      00:00:42  Thu Feb 25 09:10:53
6774      lys_NVE_20 dlebard       Running 512    11:30:58  Thu Feb 25 09:11:09

     3 active jobs :  129 of  248 hosts ( 52.02 %)

WAITING JOBS------------------------
JOBID     JOBNAME    USERNAME      STATE   CORE  WCLIMIT    QUEUETIME
================================================================================

WAITING JOBS WITH JOB DEPENDENCIES---
```

Longhorn Queue Structure



SGE Batch Environment Queues

Queue Name	*Max Runtime*	*Max Cores*	*Node Pool*
normal	6 hrs	128	All nodes
long	24 hrs	128	All nodes
largemem	8 hrs	128	16 Large memory nodes
devel	1 hrs	32	8 Nodes
request	---	---	special requests

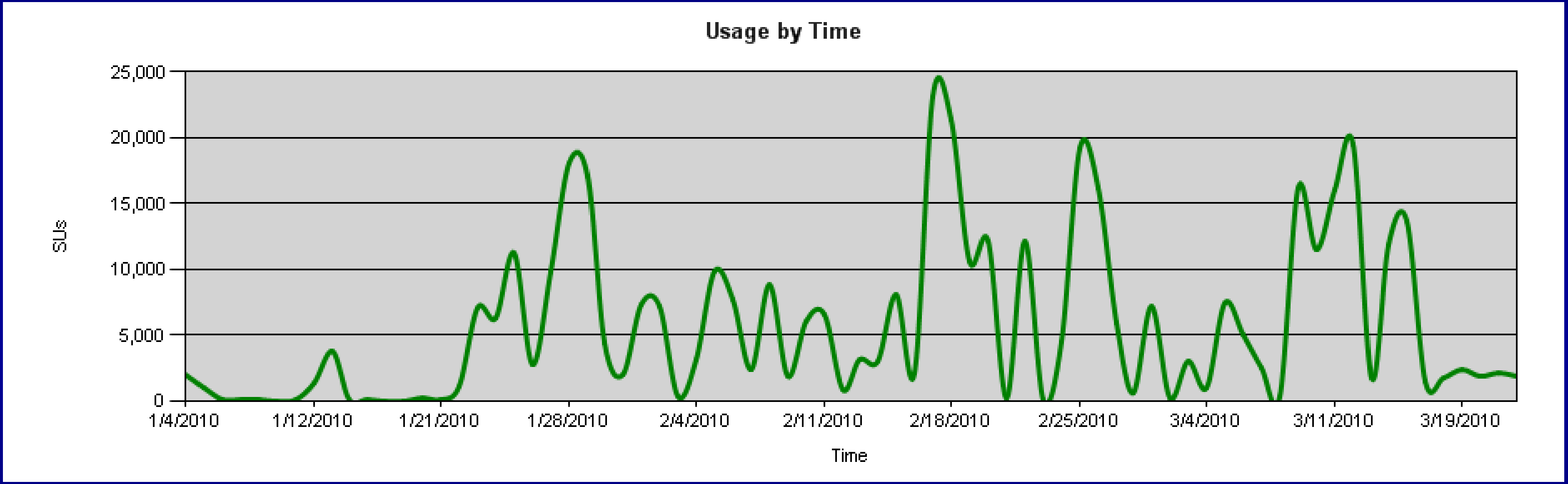


Project Types

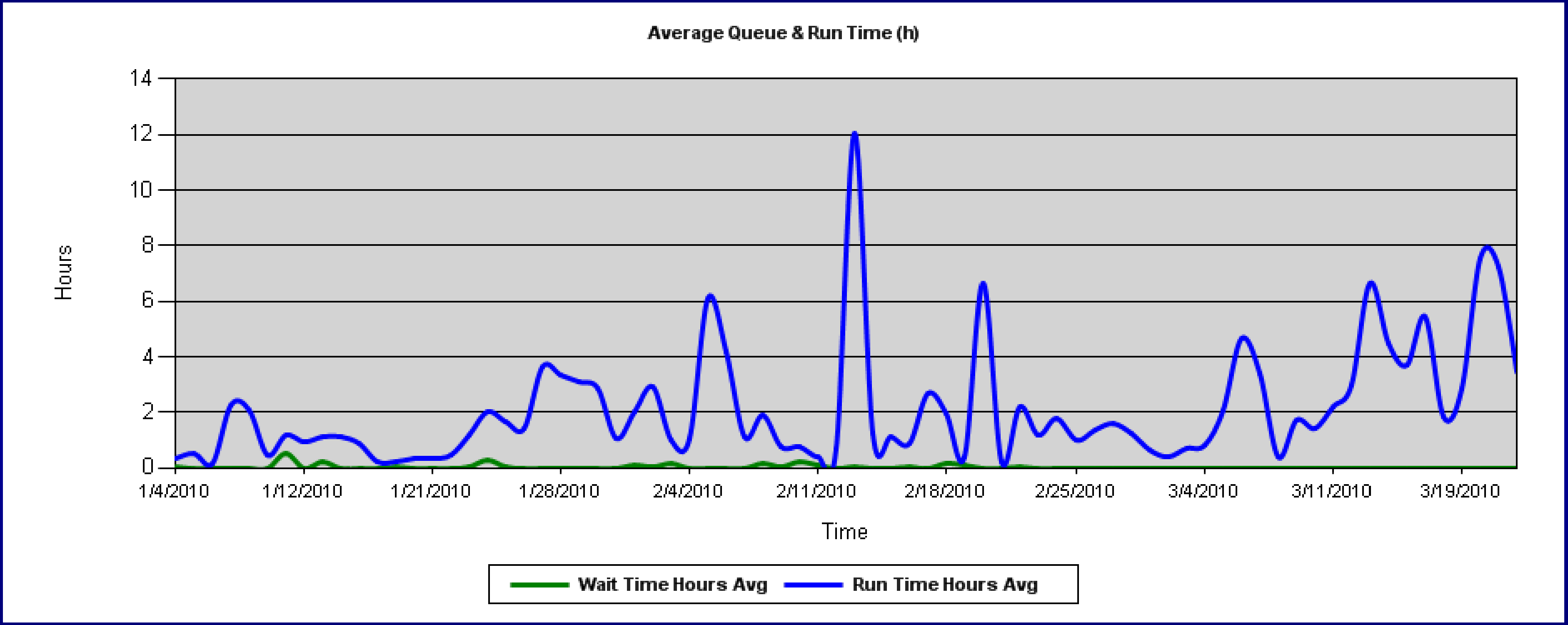
Type	*Purpose*	*Special Environment Modifications*
vis	Visualization jobs	
data	Data Analysis jobs	
gpgpu	GPGPU jobs	disables X server
hpc	HPC jobs	

```
qsub -q normal -P vis
```

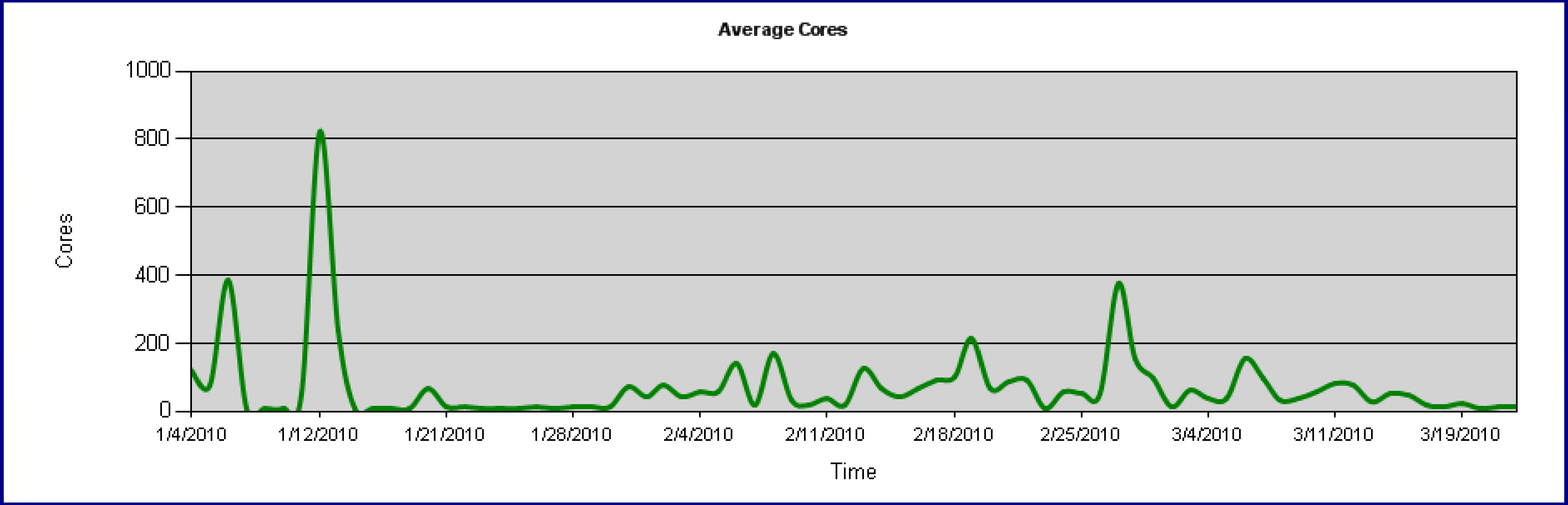
Longhorn Usage



Longhorn Usage

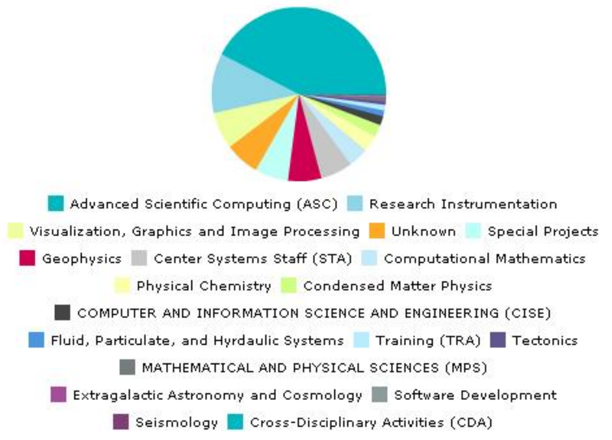


Longhorn Usage

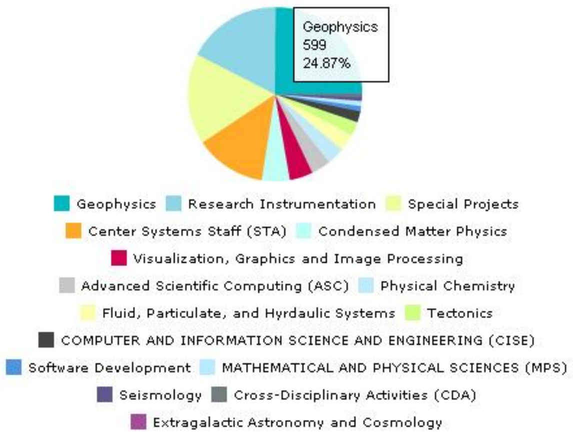


Longhorn Usage

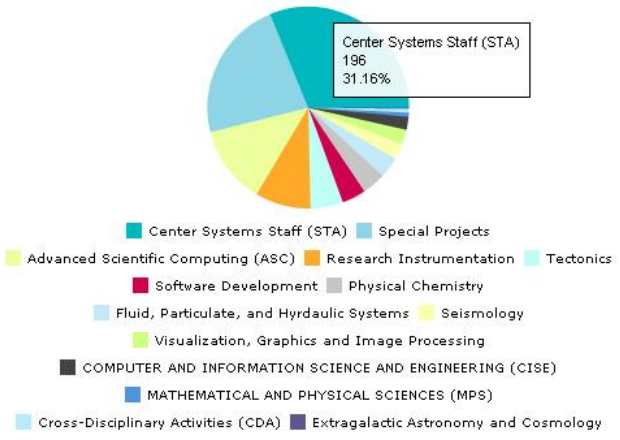
Field of Science Since Production



Field of Science Last 30 Days



Field of Science Last 7 Days



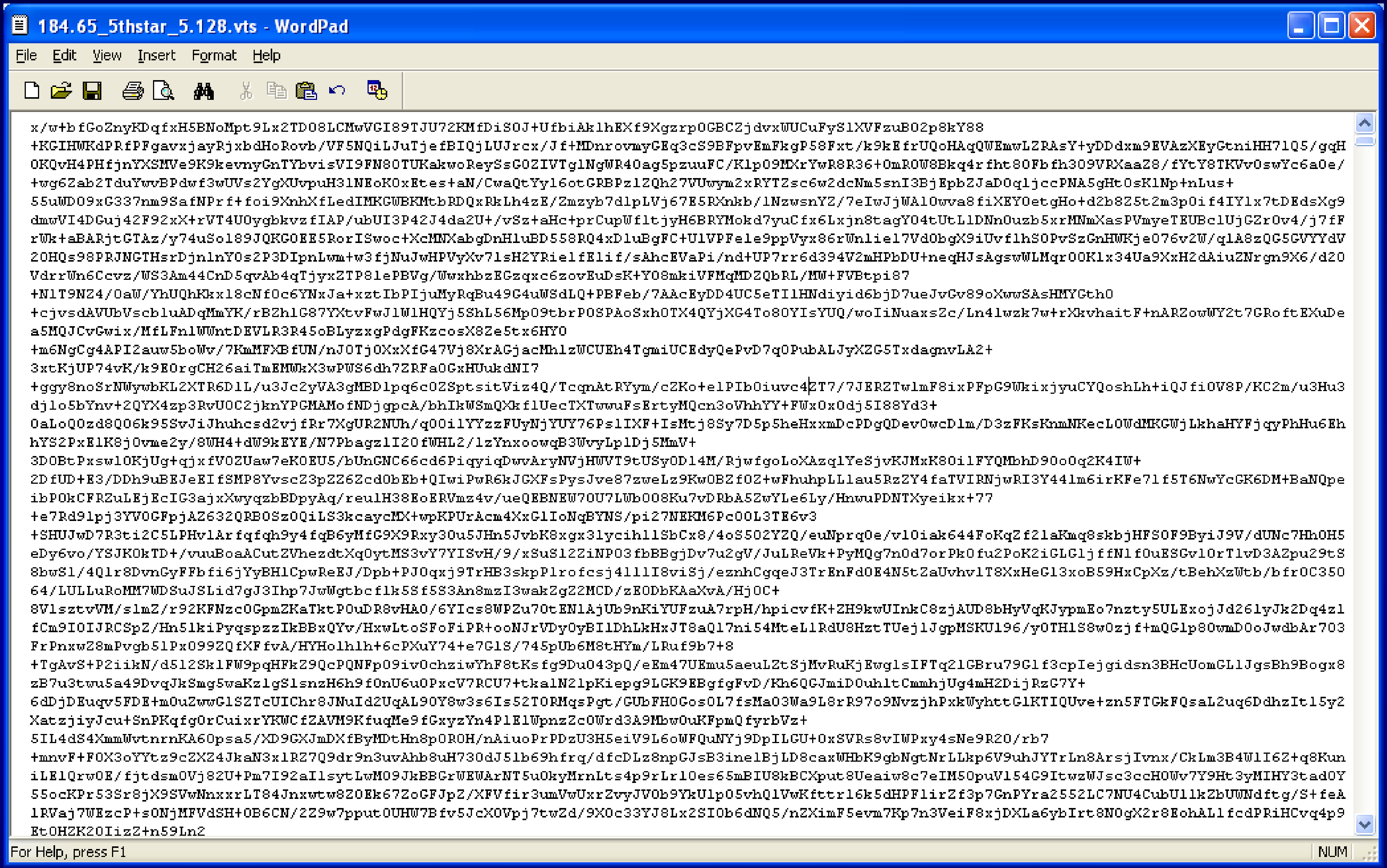
Sampling of Current Projects

- Computational Study of Earth and Planetary Materials
- Simulation of Quantum Systems
- Visualization and Analysis of Turbulent Flow
- A probabilistic Molecular Dynamics Optimized for the GPU
- Visualization of Nano-Microscopy
- MURI on Biologically-Inspired Autonomous Sea Vehicles: Towards a Mission Configurable Stealth Underwater Batoid
- Adaptive Multiscale Simulations

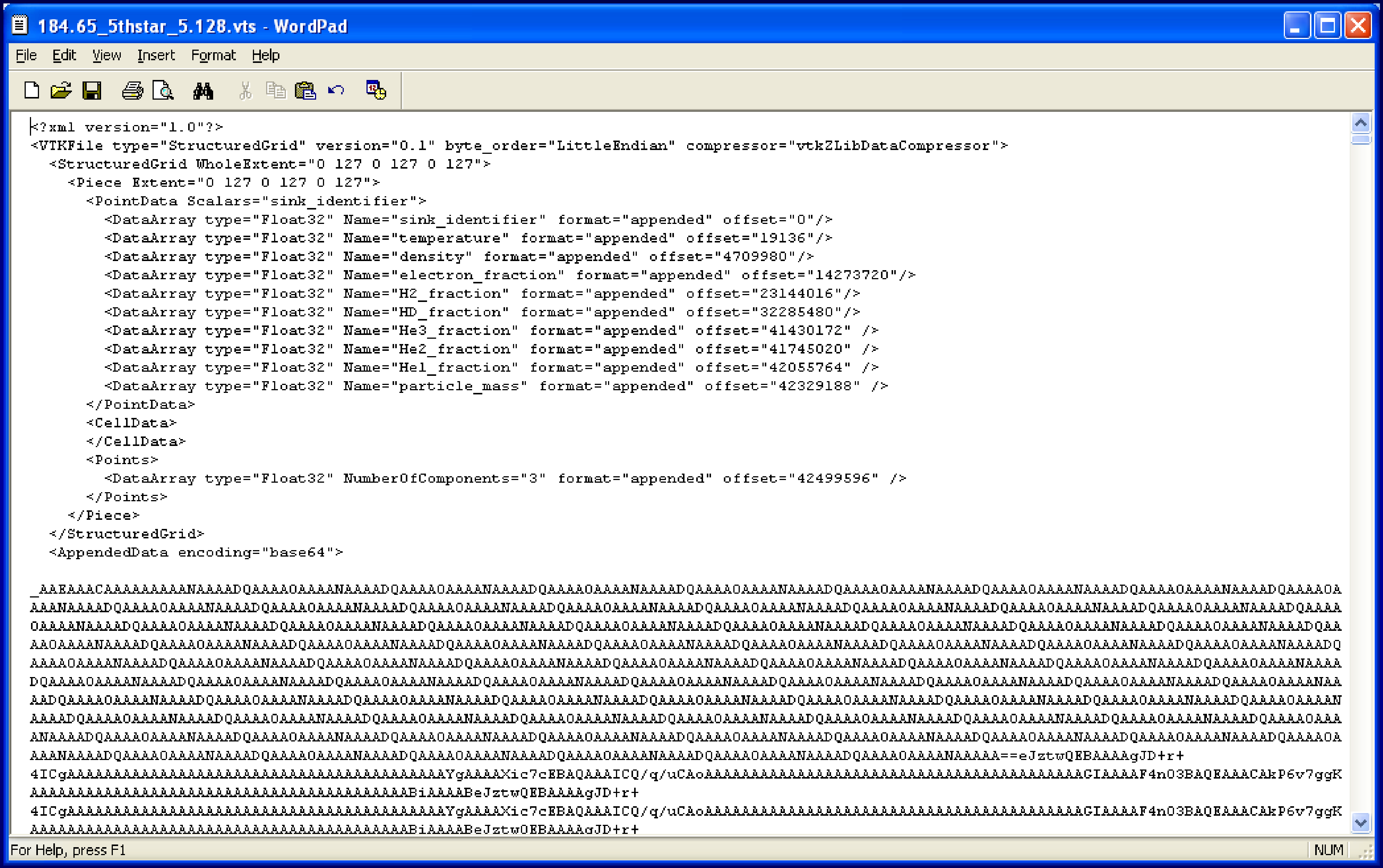
Scientific Visualization

“The purpose of computing is insight not numbers.”

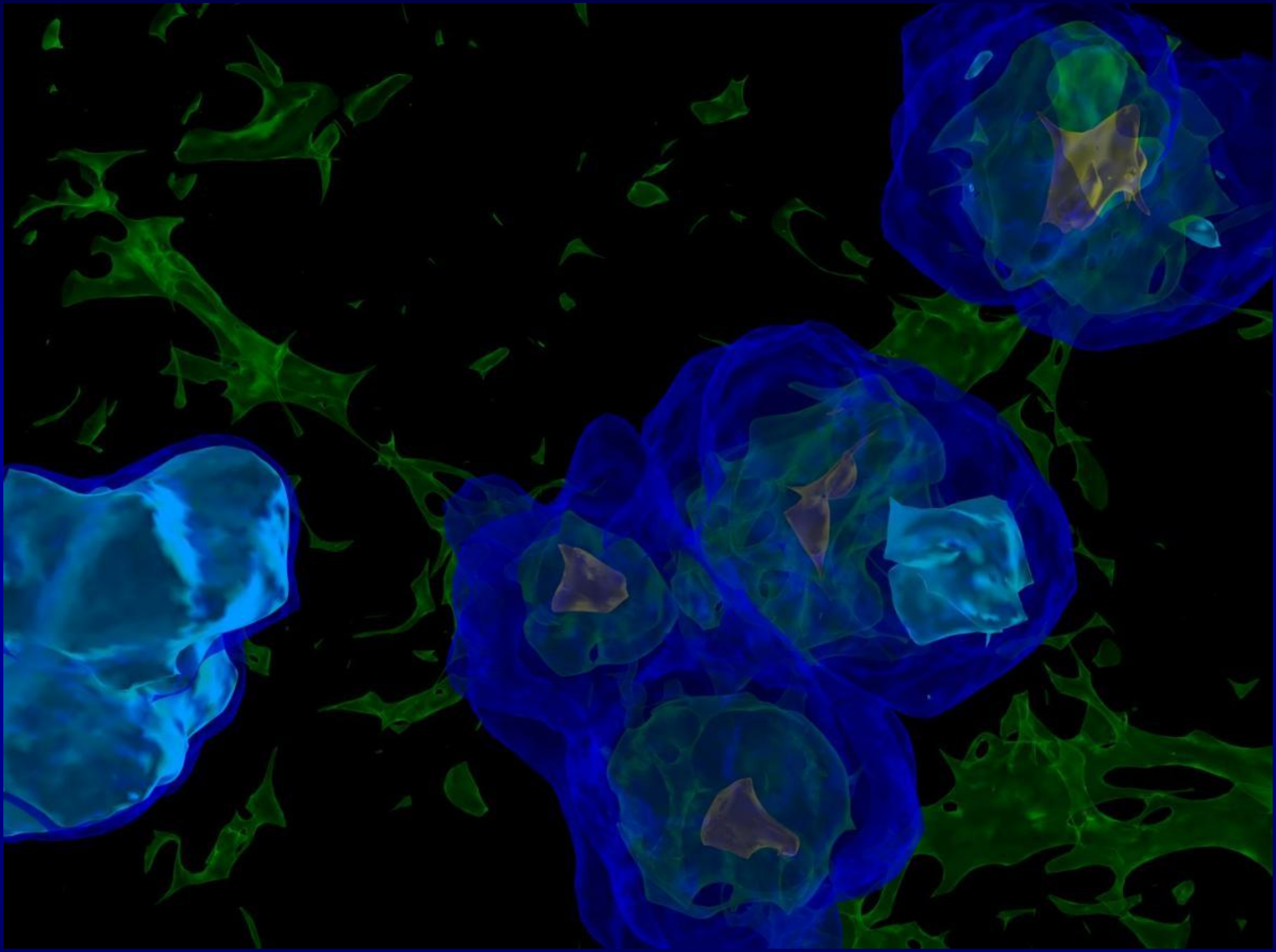
-- R. W. Hamming (1961)



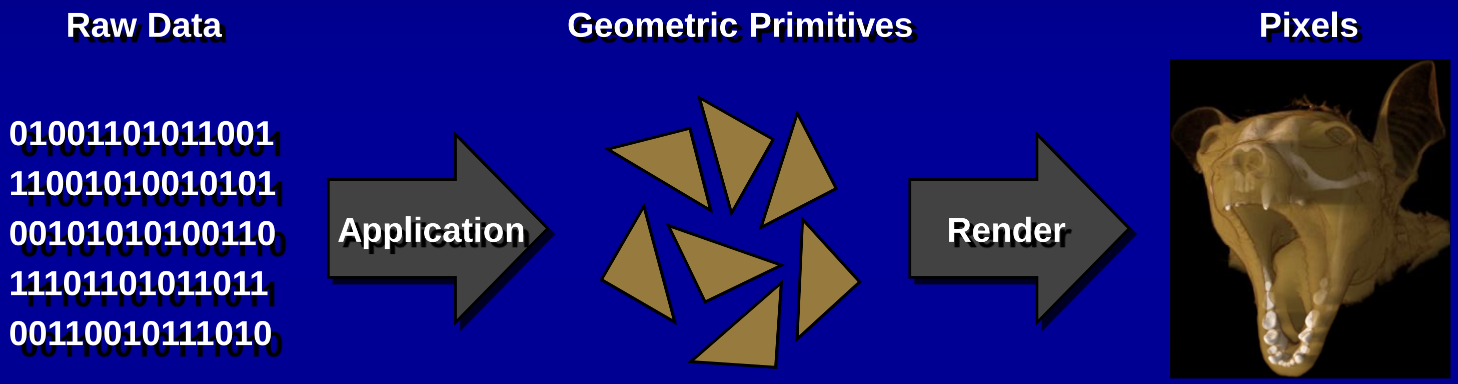
```
x/w+bfGoZnyKDqfxH5BNoMpt9Lx2TD08LCMwVGI89TJU72KMfDiS0J+UfbiAk1hEXf9Xgzrp0GBCZjdvxWUCuFyS1XVFzuB02p8kY88
+KGIHWKdPRfPFgavxjayRjxbdHoRovb/VF5NQiLJuTjefBIQjLUJrcx/Jf+MDnrovmyGEq3cS9BFpvEmFkgP58Fxt/k9kEfrUQoHAqQWEmwLZRAsY+yDDdxm9EVAzXEyGtniHH7lQ5/gqH
0KQvH4PHfjnYXSMVe9K9kevnyGnTYbvisVI9FN80TUKakwoReySsG0ZIVTglNgWR40ag5pzuuFC/K1p09MXrYwR8R36+0mR0W8Bkq4rfht80Fbfh309VRXaaZ8/fYtY8TKVv0swYc6a0e/
+wg6Zab2TduYwvBPdwf3wUVs2YgXUvpuH31NEoK0xEtes+aN/CwaQtYyl6otGRBPzlZQh27VUwym2xRYTZsc6w2dcNm5snI3BjEpbZJaD0qljccPNA5gHt0sK1Np+nLus+
55uWD09xG337nm9SafNPrf+foi9XnhXfLedIMKGWBKMtbRDQxRkLh4zE/Zmzyb7dlpLVj67E5RXnkb/lNzwsnYZ/7eIwJjWA10wva8fiXEY0etgHo+dZb8Z5t2m3p0if4IYlx7tDEdsXg9
dmwVI4DGuj42F92xX+rVT4U0ygbkvzfIAP/ubUI3P42J4da2U+/vSz+aHc+prCupWfltjyH6BRYMokd7yuCfx6Lxjn8tagY04tUtLlDNn0uzb5xrMNmXasPVmyeTEUBclUjGZr0v4/j7fF
rWk+aBARjtGTAz/y74uSol89JQKG0EE5RorISwoc+XcMNXabgDnHluBD558RQ4xDluBgFC+UlVPFele9ppVyx86rWnliel7Vd0bgX9iUvflhS0PvSzGnHWKje076v2W/qlA8zQG5GVYYdV
20HQs98PRJNGTHsrDjnlnY0s2P3DIpnLwm+w3fjNuJwHPVyXv7lsH2YRielfElif/sAhcEVaPi/nd+UP7rr6d394V2mHPbDU+neqHJsAgswWLMqr00K1x34Ua9XxH2dAiuZNrgn9X6/d20
VdrrWn6Ccvz/WS3Am44CnD5qvAb4qTjyxZTP8lePBVg/WwxhbzEGzqxc6zovEuDsK+Y08mkiVFMqMDZQbRL/MW+FVBtpi87
+N1T9NZ4/0aW/YhUQhKkxl8cNf0c6YNxJa+xztIbPIjuMyRqBu49G4uWSdLQ+PBFeb/7AAcEyDD4UC5eTIlHNdiyid6bjD7ueJvGv89oXwwSAsHMYGth0
+cjvsdAVUbVscbluADqMmYK/rBZhlG87YXtvFwJ1WlHQYj5ShL56Mp09tbrP0SPAoSxh0TX4QYjXG4To80YIsYUQ/woIiNuaxsZc/Ln4lwzk7w+rXkvhaitF+nARZowWY2t7GRoftEXuDe
a5MQJCvGwix/MfLFnlWUntDEVLR3R45oBLyzxgPdgFKzcosX8Ze5tx6HY0
+m6NgCg4API2auw5boWv/7KmMFXBfUN/nJ0Tj0XxXfG47Vj8XrAGjacMhlzWCUEh4TgmiUCEdyQePvD7q0PubALJyXZG5TxdagnvLA2+
3xtKjUP74vK/k9E0rgCH26aiTmEMWkX3wPWS6dh7ZRFa0GxHUukdNI7
+ggy8noSrNWywbKL2XTR6DlL/u3Jc2yVA3gMBDlpq6c0ZSptsitViz4Q/TcqnAtRYym/cZKo+elPIb0iuvc4ZT7/7JERZTwlmF8ixPFpG9WkixjyuCYQoshLh+iQJfi0V8P/KC2m/u3Hu3
djlo5bYnv+2QYX4zp3RvU0C2jknYPGMAMofNDjgpcA/bhIkWSmQXkflUecTXTwwuFsErtyMQcn3oVhhYY+FWx0x0dj5I88Yd3+
0aLoQ0zd8Q06k95SvJiJhuhcsd2vjfRr7XgUR2NUh/q00ilYYzzFUyNjYUY76PslIXF+IsMtj8Sy7D5p5heHxxmDcPDgQDev0wcDlm/D3zFKsKnmNKecL0WdMKGWjLkhaHYFjqyPhHu6Eh
hYS2PxElK8j0vme2y/8WH4+dW9kEYE/N7Pbagz1I20fWHL2/lzYnxoowqB3WvyLplDj5MmV+
3D0BtPxswl0KjUg+qjxfV0ZUaw7eK0EU5/bUnGNC66cd6PiqyiqDwvAryNVjHWVT9tUSy0Dl4M/RjwfgoLoXAzqlYeSjvKJMxK80ilFYQMbhD90o0q2K4IW+
2DfUD+E3/DDh9uBEJeEIfSMP8YvscZ3pZZ6Zcd0bEb+QIwiPwR6kJGXFsPysJve87zweLz9Kw0BZf0Z+wFhuhpLLlau5RzZY4faTVIRNjwRI3Y44lm6irKFe7lf5T6NwYcGK6DM+BaNQpe
ibP0kCFRZuLEjEcIG3ajxXwyqzbBDpyAq/reulH38EoERVmz4v/ueQEBNEW70U7LWb008Ku7vDRbA5ZwYLe6Ly/HnwuPDNTXyeikx+77
+e7Rd9lpj3YV0GFpjAZ632QRB0Sz0QiLS3kcaycMX+wpKPUrAcm4XxGlIoNqBYNS/pi27NEKM6Pc00L3TE6v3
+SHUJwD7R3ti2C5LPHvlArfqfqh9y4fqB6yMfG9X9Rxy30u5JHn5JvbK8xgx3lycihllSbCx8/4oS502YZQ/euNprq0e/vl0iak644FoKqZf2laKmq8skbjHFS0F9ByiJ9V/dUNc7Hh0H5
eDy6vo/YSJK0kTD+/vuuBoaACutZVhezdtXq0ytMS3vY7YISvH/9/xSuSl2ZiNP03fbBBgjDv7u2gV/JuLReVk+PyMQg7n0d7orPk0fu2PoK2iGLGljffNlf0uESGvl0rTlvD3AZpu29tS
8bwSl/4Qlr8DvnGyFFbfi6jYyBHlCpwReEJ/Dpb+PJ0qxj9TrHB3skpPlrofcsj4lllI8viSj/eznhCgqeJ3TrEnFd0E4N5tZaUvhvlT8XxHeGl3xoB59HxCpXz/tBehXzWtb/bfr0C350
64/LULLuRoMM7WDSuJSLid7gJ3Ihp7JwWgtbcflk5Sf5S3An8mzI3wakZgZ2MCD/zE0DbKAaXvA/Hj0C+
8VlsztvVM/slmZ/r92KFNzc0GpmZKaTktP0uDR8vHA0/6YIcs8WPZu70tENlAjUb9nKiYUFzuA7rpH/hpicvfK+ZH9kwUInkC8zjAUD8bHyVqKJypmEo7nzty5ULExojJd26lyJk2Dq4zl
fCm9I0IJRCSpZ/Hn5lkiPyqspzzIkBBxQYv/HxwLtoSFoFiPR+ooNJrVDy0yBIlDhLkHxJT8aQl7ni54MteLlRdU8HztTUej1JgpMSKU196/y0THlS8w0zjf+mQGlp80wmD0oJwdbAr703
FrPnxwZ8mPvgb5lPx099ZQfXFfvA/HYHolhlh+6cPXuY74+e7GlS/745pUb6M8tHYm/LRuf9b7+8
+TgAvS+P2iikN/d5l2SklFW9pqHFkZ9QcPQNFp09iv0chziwYhF8tKsfg9Du043pQ/eEm47UEmu5aeuLZtSjMvRuKjEwglsIFTq2lGBru79Glf3cpIejgidsn3BHcUomGLlJgsBh9Bogx8
zB7u3twu5a49DvqJkSmg5waKzlgSlsnzH6h9f0nU6u0PxcV7RCU7+tkalN2lpKiepg9LGK9EBgfgFvD/Kh6QGJmiD0uhltCmmhjUg4mH2DijRzG7Y+
6dDjDEuqv5FDE+m0uZwwGlSZTcUIChr8JNuId2UqAL90Y8w3s6Is52T0RMqsPgt/GUbFH0Gos0L7fsMa03Wa9L8rR97o9NvzjhPxkWyhttGlKTIQUve+zn5FTGkFQsaL2uq6DdhzItl5y2
XatzjiyJcu+SnPKqfg0rCuixrYKWCfZAVM9KfuqMe9fGxyzYn4PlElWpnzZc0Wrd3A9Mbw0uKFpmQfyrbVz+
5IL4dS4XmmWvtnrnKA60psa5/XD9GXJmDXfByMDtHn8p0R0H/nAiuoPrPDzU3H5eiV9L6oWFQuNYj9DpILGU+0xSVRs8vIWPxy4sNe9R20/rb7
+mnvF+F0X3oYYtz9cZX24JkaN3xlRZ7Q9dr9n3uvAhb8uH730dJ5lb69hfrq/dfcDLz8npGJsB3inelBjLD8caxWHbK9gbNgtNrLLkp6V9uhJYTrLn8ArsjIvnx/CkLm3B4WlI6Z+q8Kun
iLElQrw0E/fjtdsm0Vj82U+Pm7I92aIlsytLwM09JkBBGrWEWArNT5u0kyMrnLts4p9rLrl0es65mBIU8kBCXput8Ueaiw8c7eIM50puVl54G9ItwzWJsc3ccH0Wv7Y9Ht3yMIHY3tad0Y
55ocKPr53Sr8jX9SVwNnxxrLT84Jnxwtw8Z0Ek67ZoGFJpZ/XFVfir3umVwUxrZvyJV0b9YkUlp05vhQlVwKfttrl6k5dHPFlirZf3p7GnPYra2552LC7NU4CubUllkZbUWNdftg/S+feA
lRVaj7WEzcP+s0NjMFVdSH+0B6CN/2Z9w7pput0UHW7Bfv5JcX0Vpj7twZd/9X0c33YJ8Lx2SI0b6dNQ5/nZXimF5evm7Kp7n3VeiF8xjDXLa6ybIrt8N0gX2r8EohALlfcdPRiHCvq4p9
Et0HZK20IizZ+n59Ln2
```



```
<?xml version="1.0"?>
<VTKFile type="StructuredGrid" version="0.1" byte_order="LittleEndian" compressor="vtkZLibDataCompressor">
  <StructuredGrid WholeExtent="0 127 0 127 0 127">
    <Piece Extent="0 127 0 127 0 127">
      <PointData Scalars="sink_identifier">
        <DataArray type="Float32" Name="sink_identifier" format="appended" offset="0"/>
        <DataArray type="Float32" Name="temperature" format="appended" offset="19136"/>
        <DataArray type="Float32" Name="density" format="appended" offset="4709980"/>
        <DataArray type="Float32" Name="electron_fraction" format="appended" offset="14273720"/>
        <DataArray type="Float32" Name="H2_fraction" format="appended" offset="23144016"/>
        <DataArray type="Float32" Name="HD_fraction" format="appended" offset="32285480"/>
        <DataArray type="Float32" Name="He3_fraction" format="appended" offset="41430172" />
        <DataArray type="Float32" Name="He2_fraction" format="appended" offset="41745020" />
        <DataArray type="Float32" Name="Hel_fraction" format="appended" offset="42055764" />
        <DataArray type="Float32" Name="particle_mass" format="appended" offset="42329188" />
      </PointData>
      <CellData>
      </CellData>
      <Points>
        <DataArray type="Float32" NumberOfComponents="3" format="appended" offset="42499596" />
      </Points>
    </Piece>
  </StructuredGrid>
  <AppendedData encoding="base64">

_AAEAAACAAAAAAAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOA
AAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAA
OAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAA
AAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQ
AAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAA
DQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAA
AADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAAN
AAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAA
ANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOA
AAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAADQAAAAOAAAANAAAAA==eJztwQEBAAAAgJD+r+
4ICgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAYgAAAAXic7cEBAQAAAICQ/q/uCAoAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGIAAAAF4n03BAQEAAACAkP6v7ggK
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABiAAAABeJztwQEBAAAAgJD+r+
4ICgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAYgAAAAXic7cEBAQAAAICQ/q/uCAoAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGIAAAAF4n03BAQEAAACAkP6v7ggK
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABiAAAABeJztwQEBAAAAgJD+r+
```



Visualization Allows Us to "See" the Science



Raw Data

01001101011001
11001010010101
00101010100110
11101101011011
00110010111010

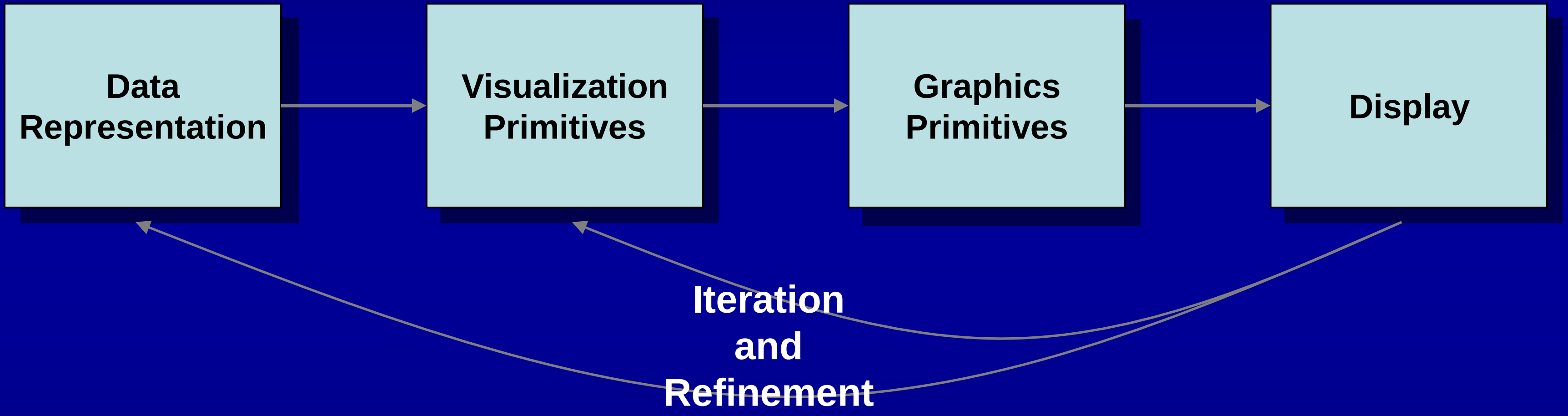
Application

Geometric Primitives

Render

Pixels

Getting from Data to Insight



Data Representation

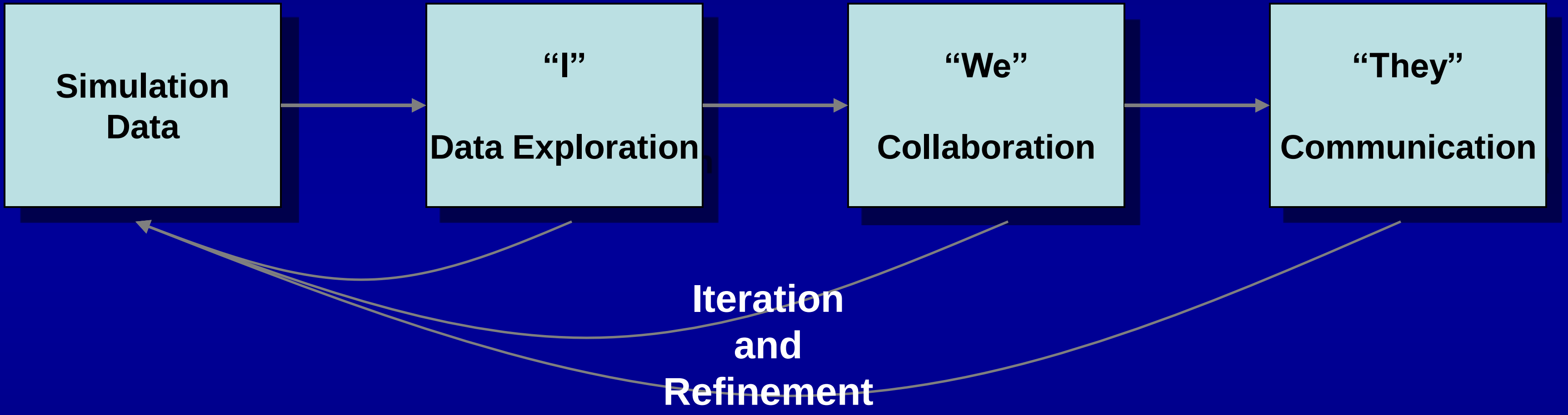
Visualization Primitives

Graphics Primitives

Display

Iteration and Refinement

“I, We, They” Development Path



Simulation Data

“I”

Data Exploration

“We”

Collaboration

“They”

Communication

Iteration
and
Refinement

Visualization Process Summary

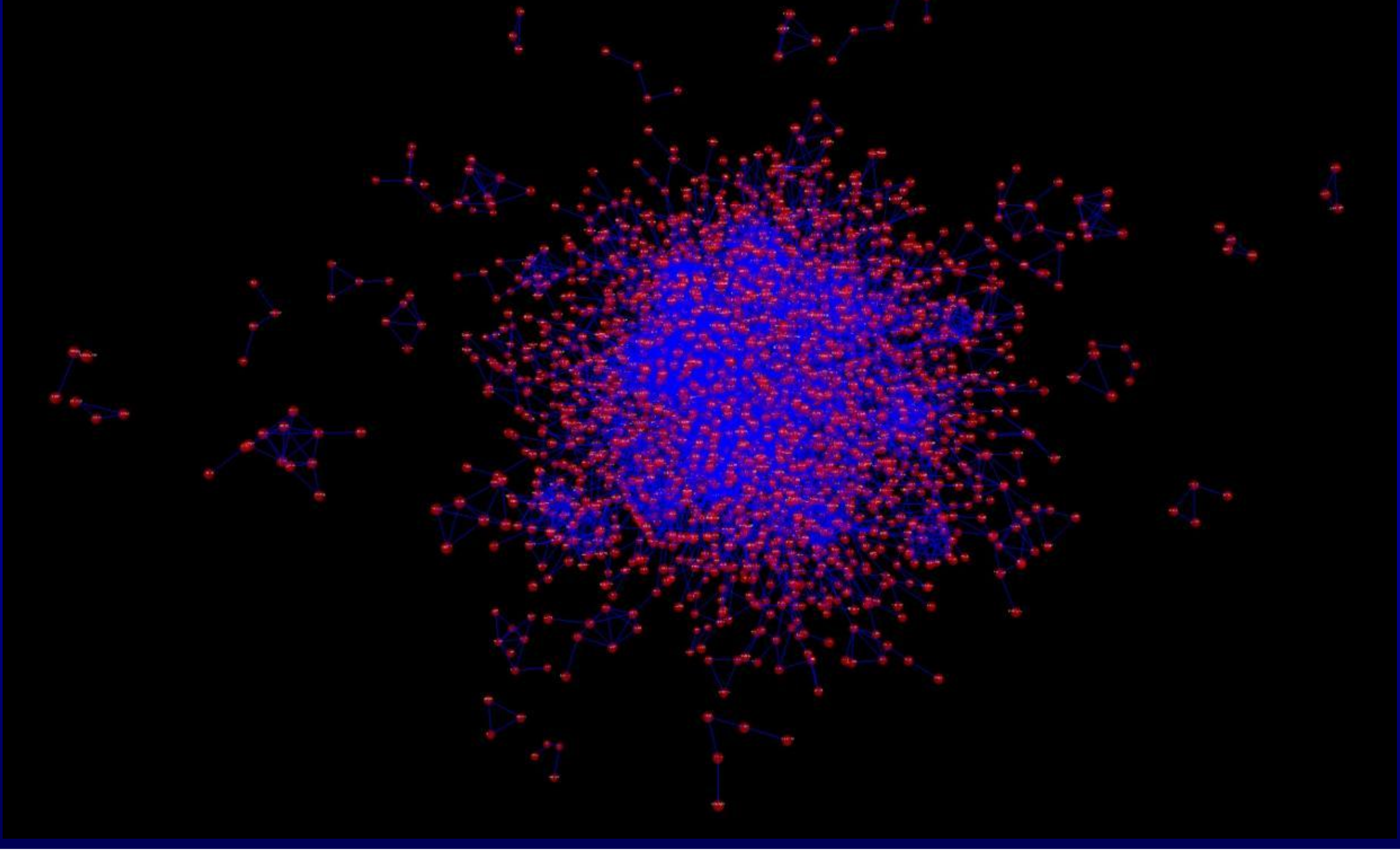
- The primary goal of visualization is ***insight***
- A picture is worth not just 1000 words,
  but potentially tera- or peta-bytes of data
- Larger datasets demand not just visualization, but advanced visualization resources and techniques
- Visualization system technology improves with advances in GPUs and LCD technology
- Visualization software slower to adapt

Types of Input Data

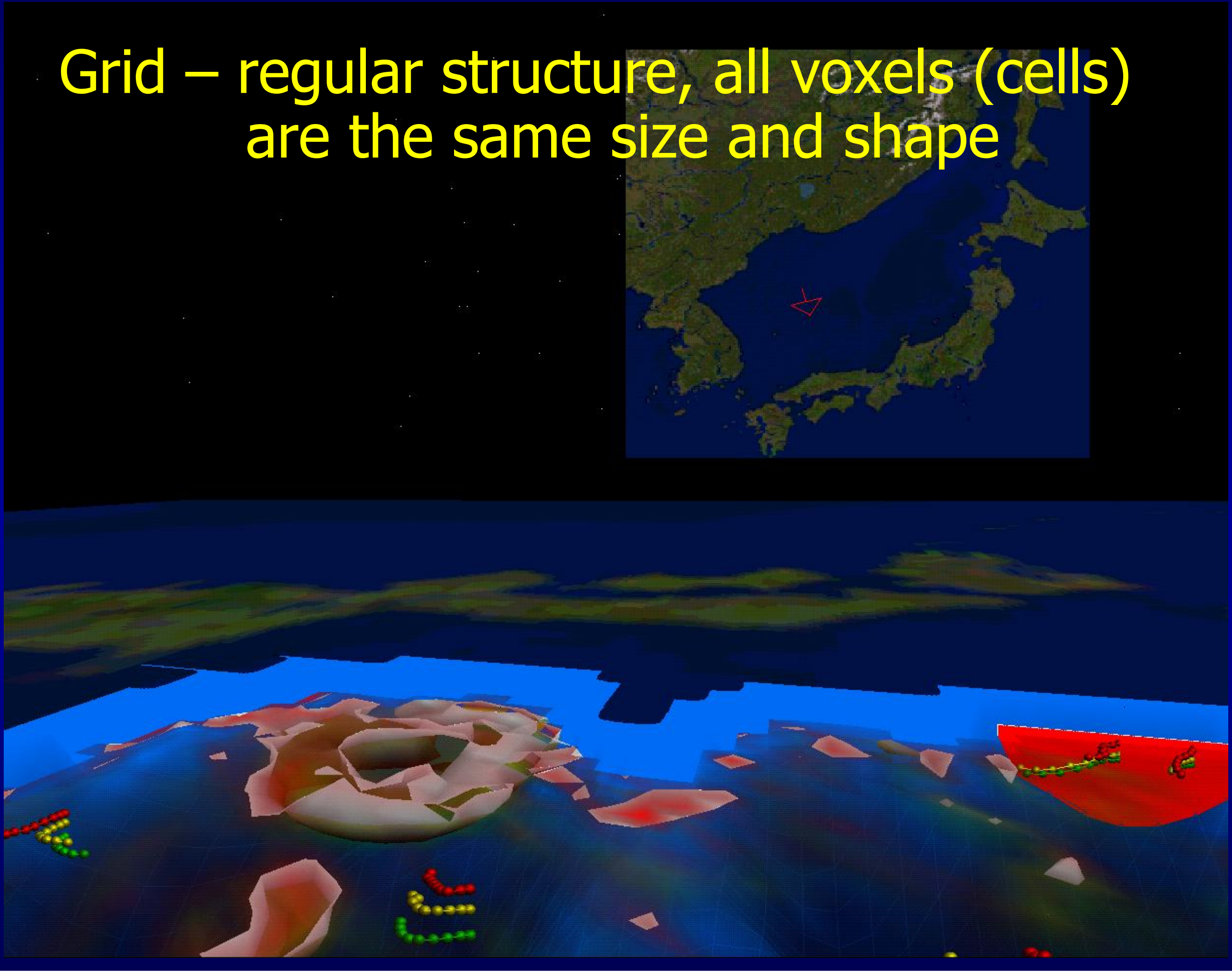
- Point / Particle
  - N-body simulation
- Regular grid
  - Medical scan
- Curvilinear grid
  - Engineering model
- Unstructured grid
  - Extracted surface

Types of Input Data

Point – scattered values with no defined structure



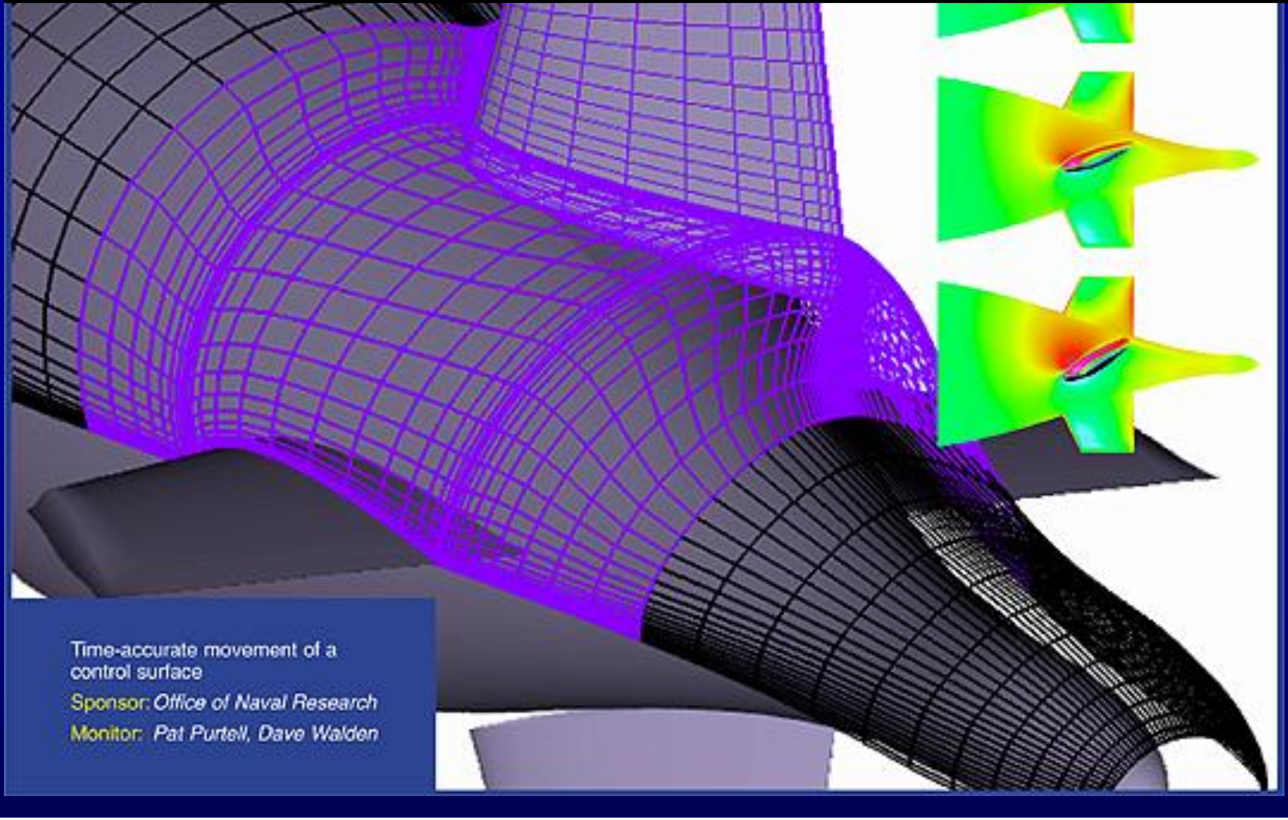
Types of Input Data



Grid – regular structure, all voxels (cells) are the same size and shape

Types of Input Data

Curvilinear – regularly grided mesh
shaping function applied



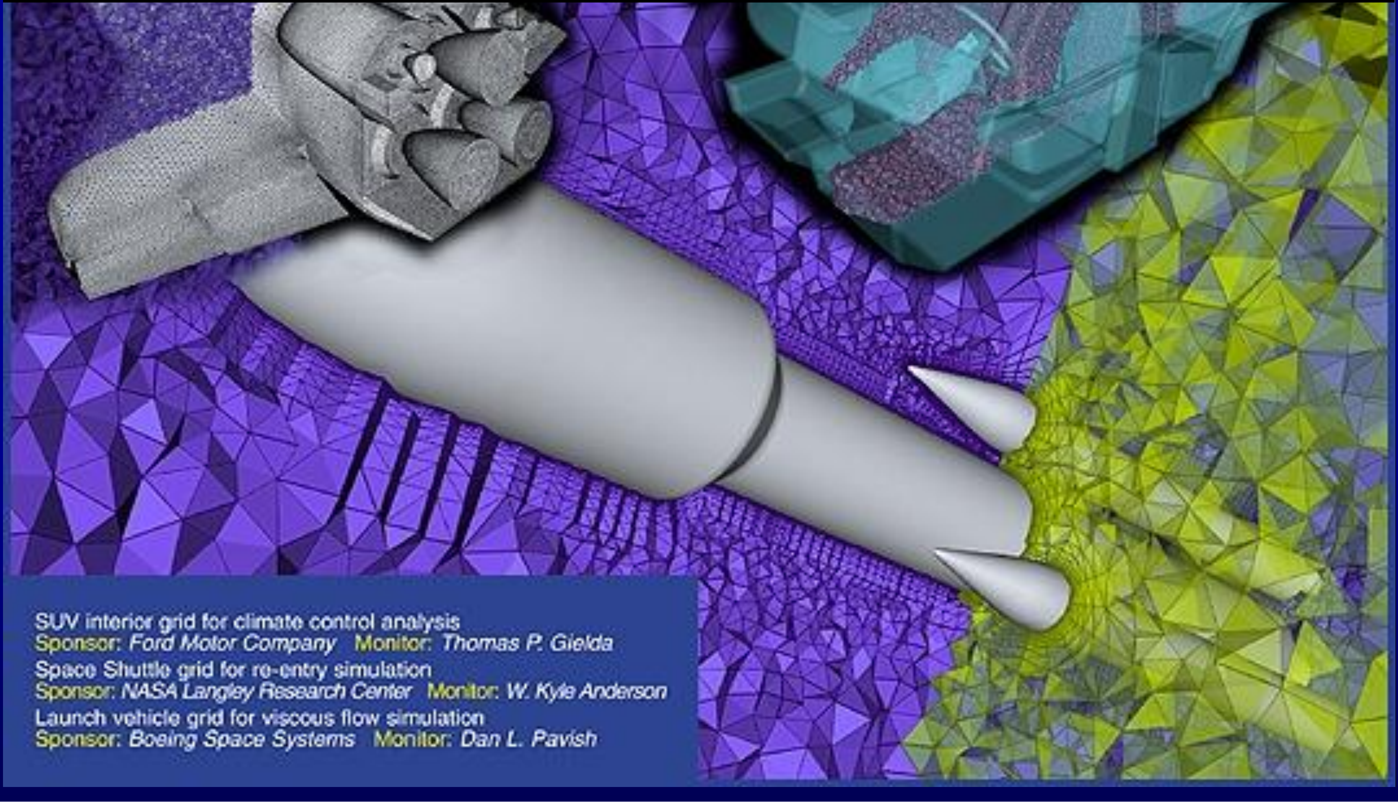
Time-accurate movement of a control surface

Sponsor: *Office of Naval Research*

Monitor: *Pat Purtell, Dave Walden*

Types of Input Data

Unstructured grid – irregular mesh typically composed of tetrahedra, prisms, pyramids, or hexahedra.



SUV interior grid for climate control analysis
Sponsor: Ford Motor Company Monitor: Thomas P. Gielda
Space Shuttle grid for re-entry simulation
Sponsor: NASA Langley Research Center Monitor: W. Kyle Anderson
Launch vehicle grid for viscous flow simulation
Sponsor: Boeing Space Systems Monitor: Dan L. Pavish

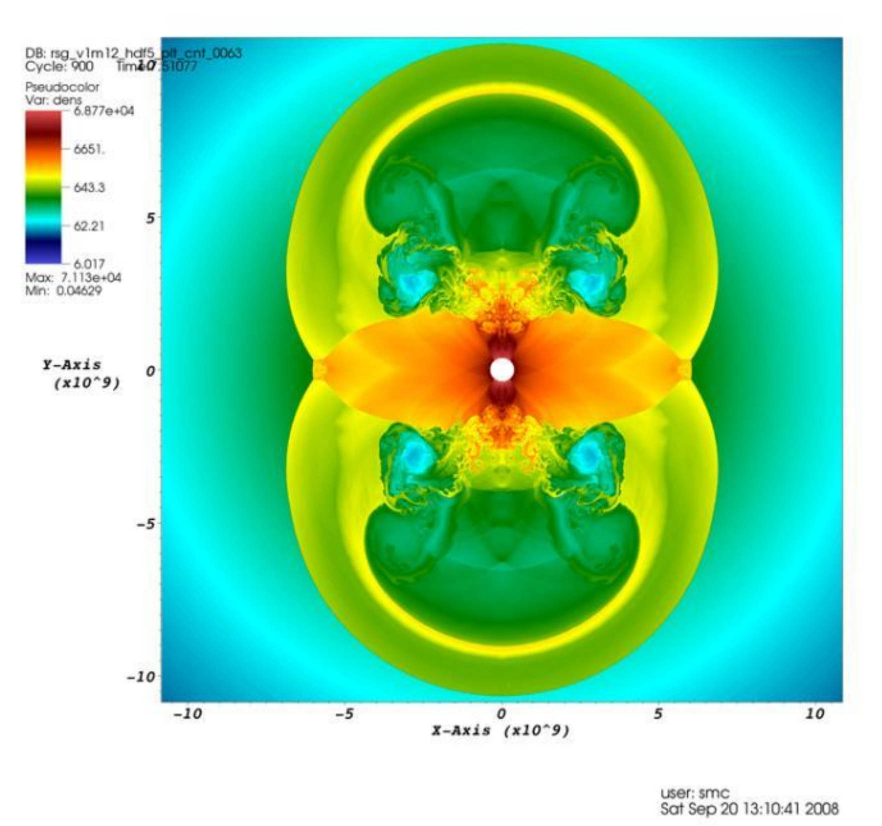
Visualization Operations

- Surface Shading (Pseudocolor)
- Isosufacing (Contours)
- Volume Rendering
- Clipping Planes
- Streamlines

Surface Shading (Pseudocolor)

Given a scalar value at a point on the surface and a color map, find the corresponding color (and opacity) and apply it to the surface point.

Most common operation, often combined with other ops

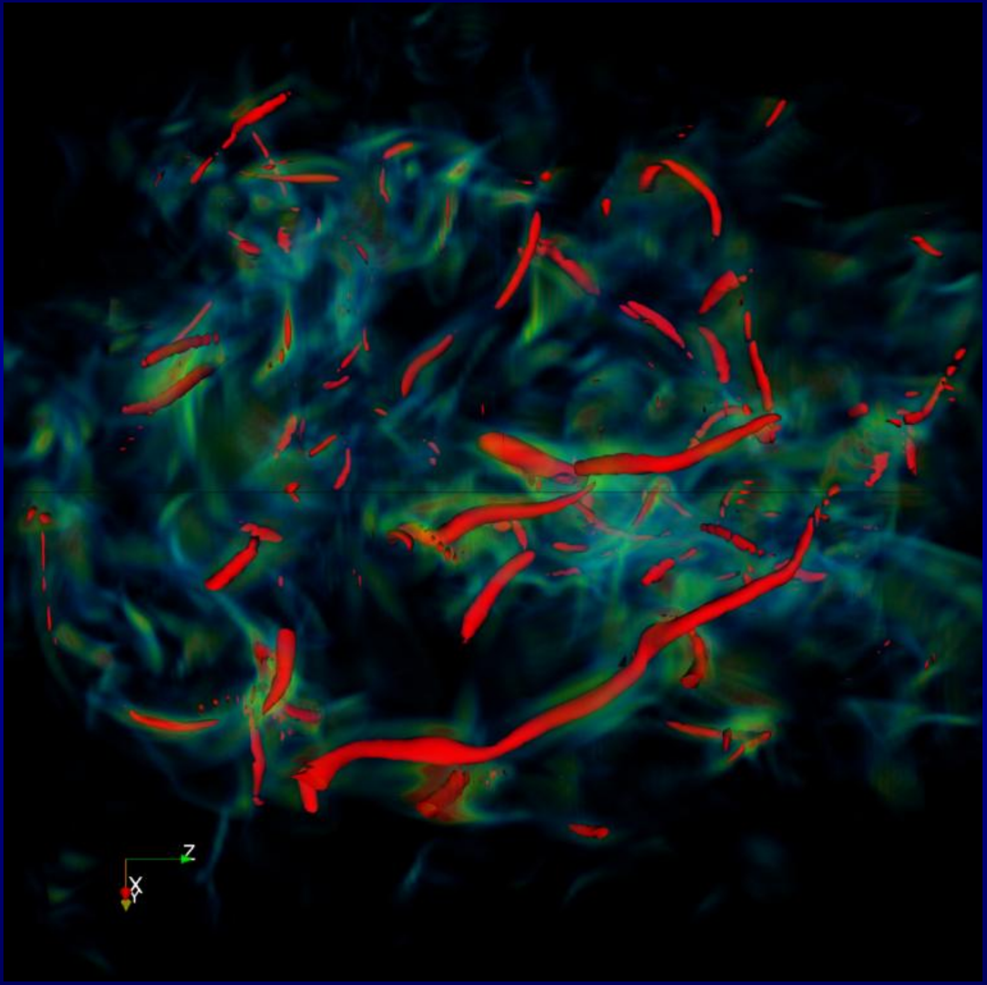


Isosurfaces (Contours)

Plot the surface for a given scalar value.

Good for showing known values of interest

Good for sampling through a data range

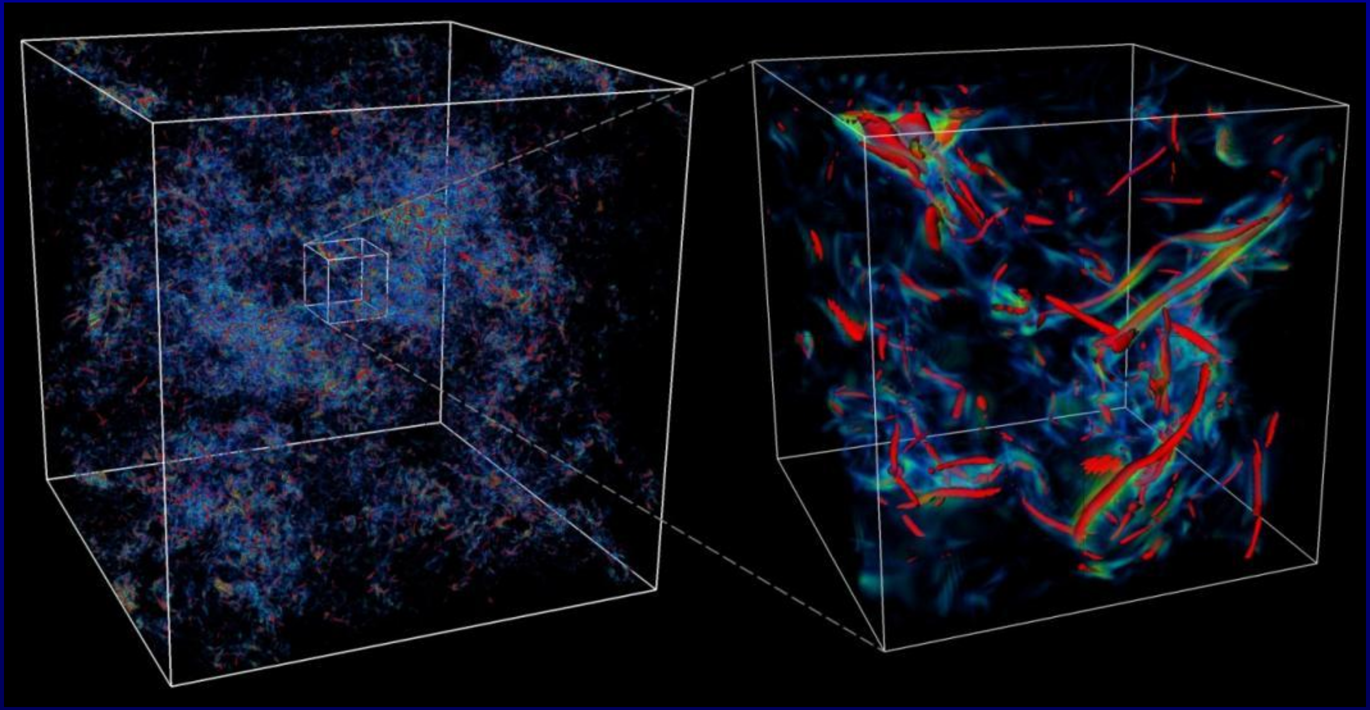


Volume Rendering

Expresses how light travels through a volume

Color and opacity controlled by transfer function

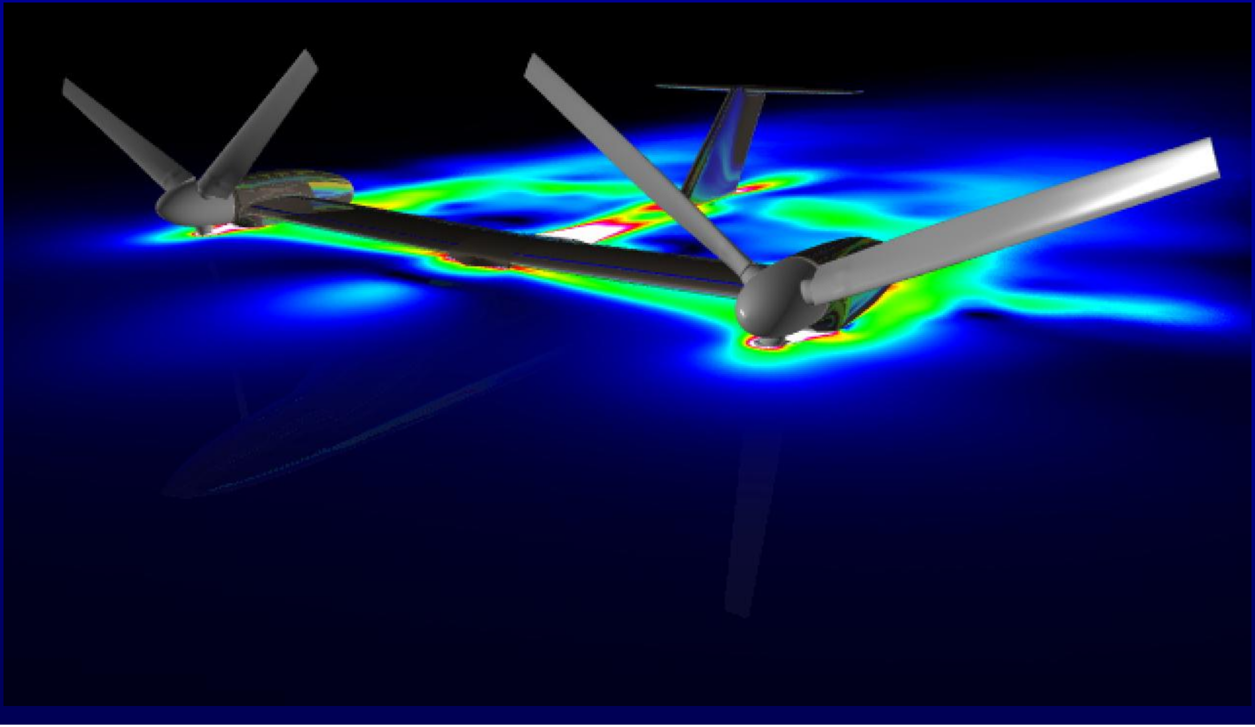
Smoother transitions than isosurfaces



Clipping / Slicing Planes

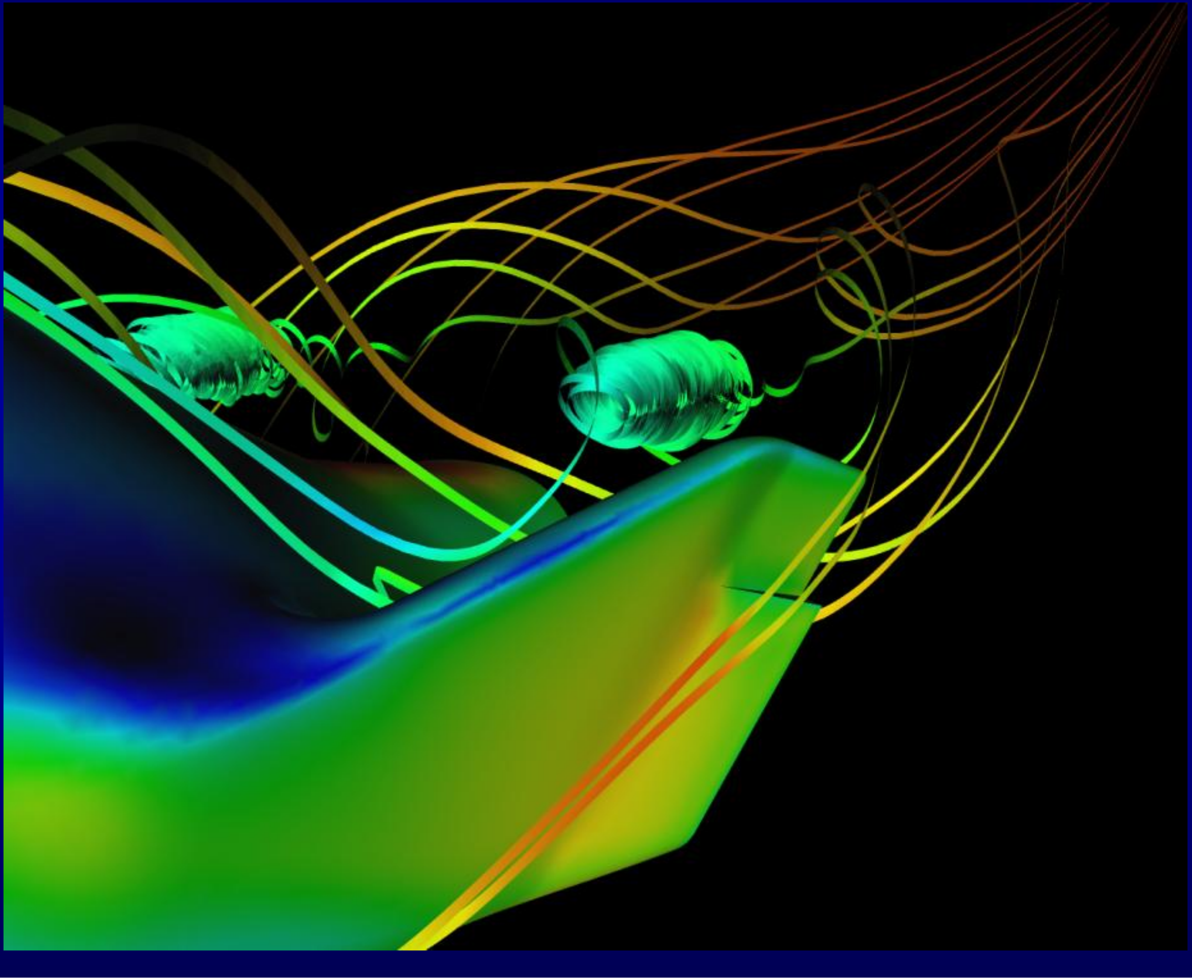
Extract a plane from the data to show features

Hide part of dataset to expose features



Particle Traces (Streamlines)

Given a vector field, extract a trace that follows that trajectory defined by the vector.



$P_{new} = P_{current} + V_P \Delta t$

Streamlines – trace in space

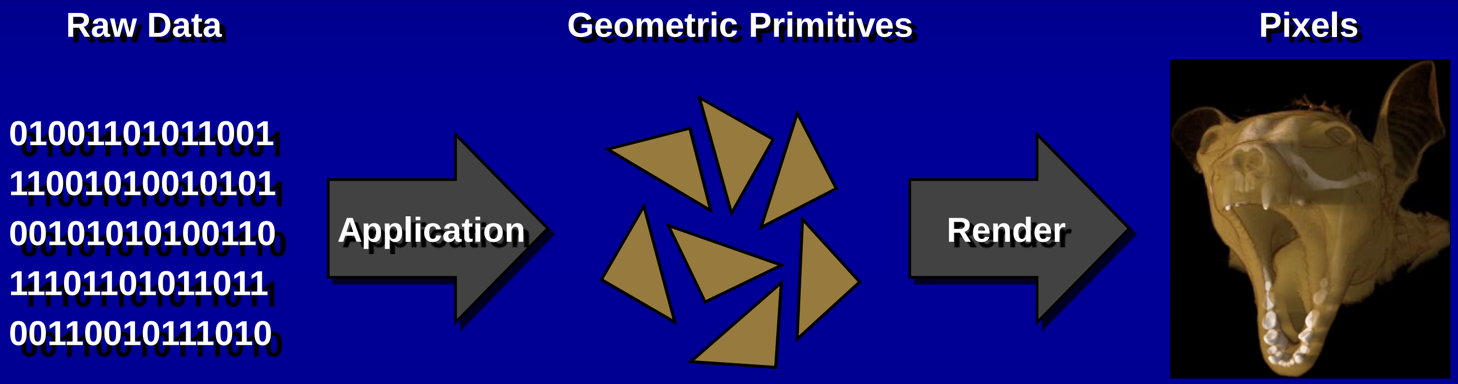
Pathlines – trace in time

Visualization Resources

- Personal machines
  - Most accessible, least powerful
- Projection systems
  - Seamless image, high purchase and maintenance costs
- Tiled-LCD displays
  - Lowest per-pixel costs, bezels divide image
- Remote visualization
  - Access to high-performance system, latency can affect user experience

Visualization Challenges

Visualization Allows Us to “See” the Science



Raw Data

01001101011001
11001010010101
00101010100110
11101101011011
00110010111010

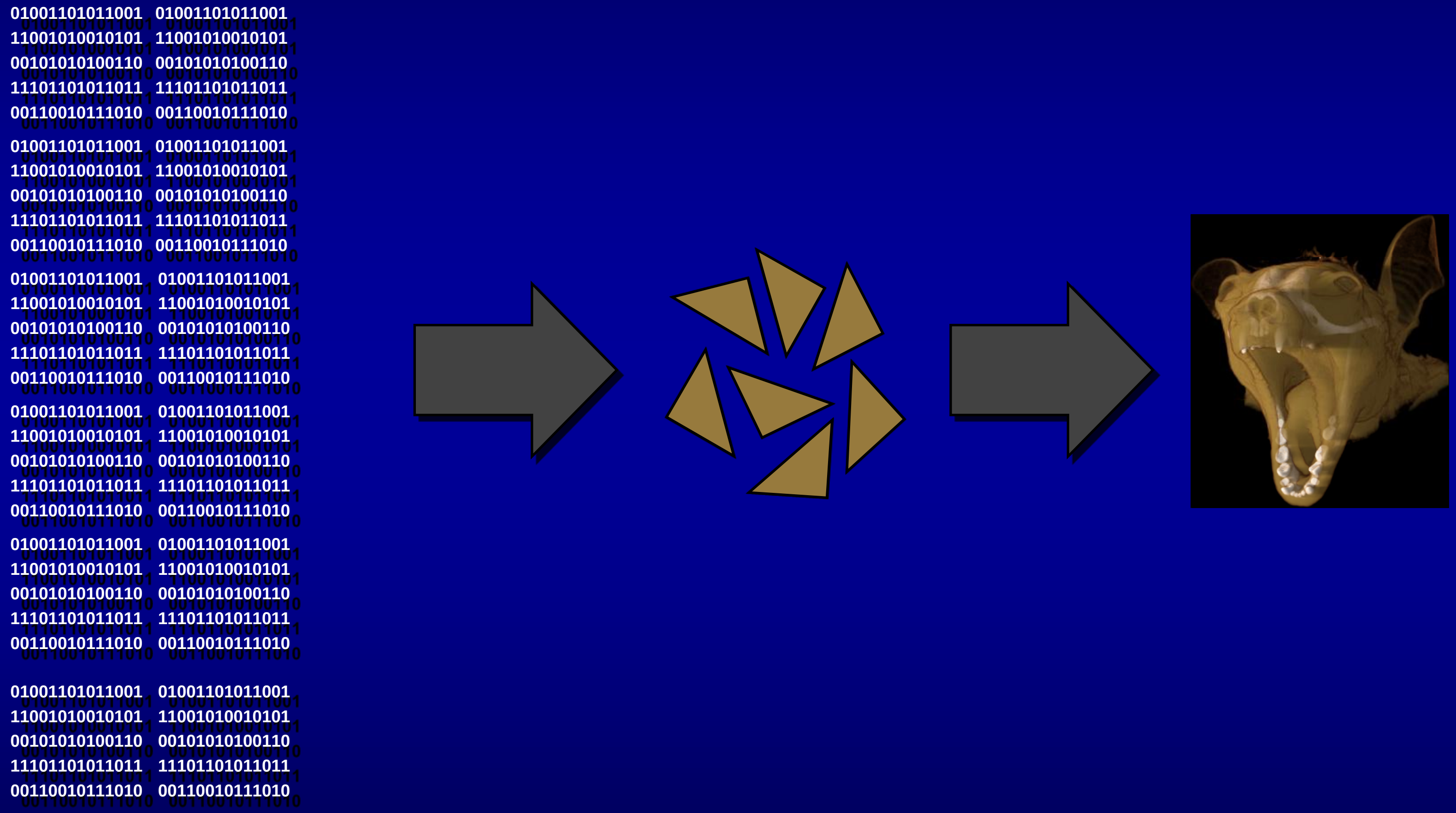
Application

Geometric Primitives

Render

Pixels

But what about large, distributed data?



01001101011001 01001101011001
11001010010101 11001010010101
00101010100110 00101010100110
11101101011011 11101101011011
00110010111010 00110010111010

01001101011001 01001101011001
11001010010101 11001010010101
00101010100110 00101010100110
11101101011011 11101101011011
00110010111010 00110010111010

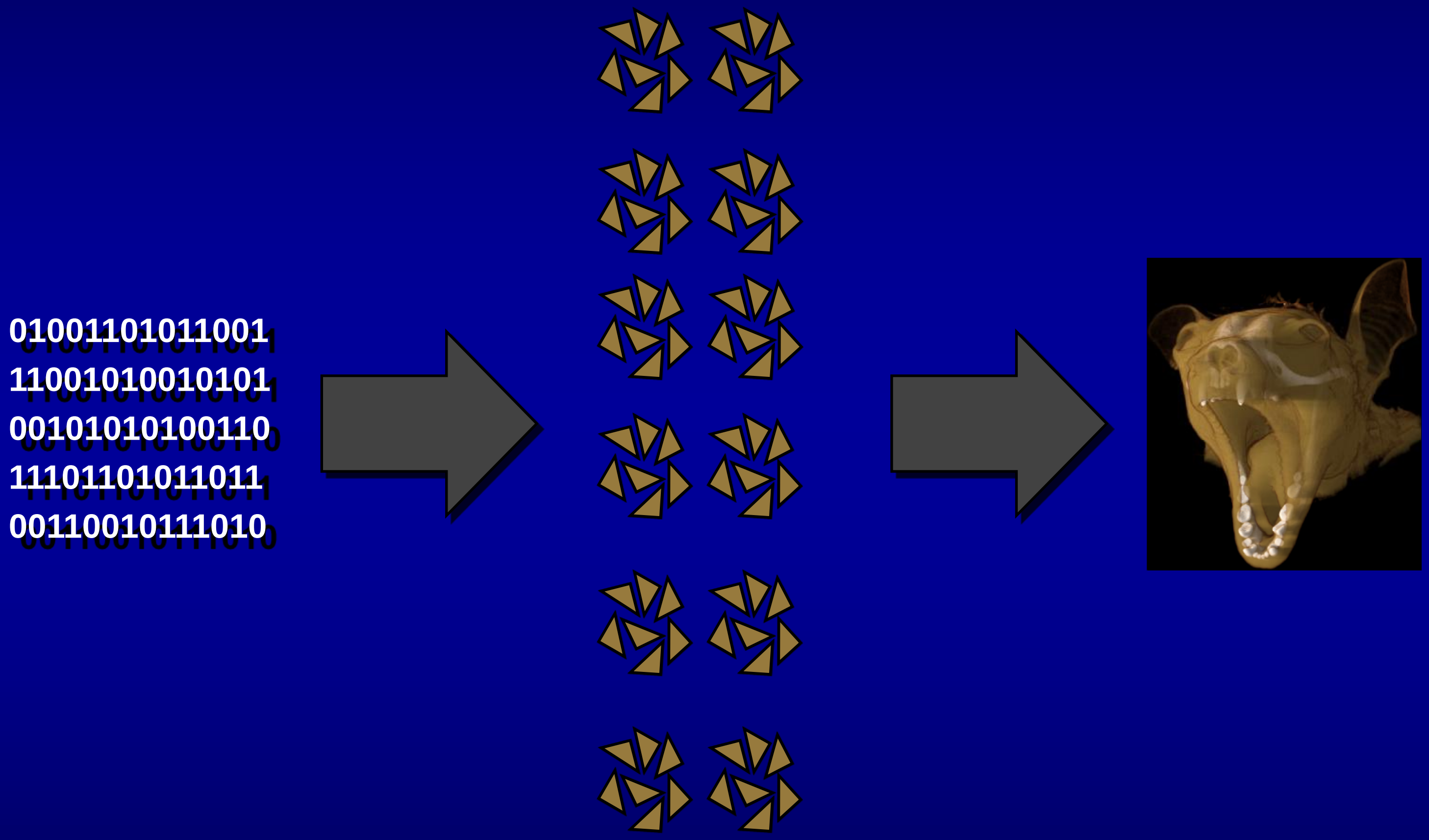
01001101011001 01001101011001
11001010010101 11001010010101
00101010100110 00101010100110
11101101011011 11101101011011
00110010111010 00110010111010

01001101011001 01001101011001
11001010010101 11001010010101
00101010100110 00101010100110
11101101011011 11101101011011
00110010111010 00110010111010

01001101011001 01001101011001
11001010010101 11001010010101
00101010100110 00101010100110
11101101011011 11101101011011
00110010111010 00110010111010

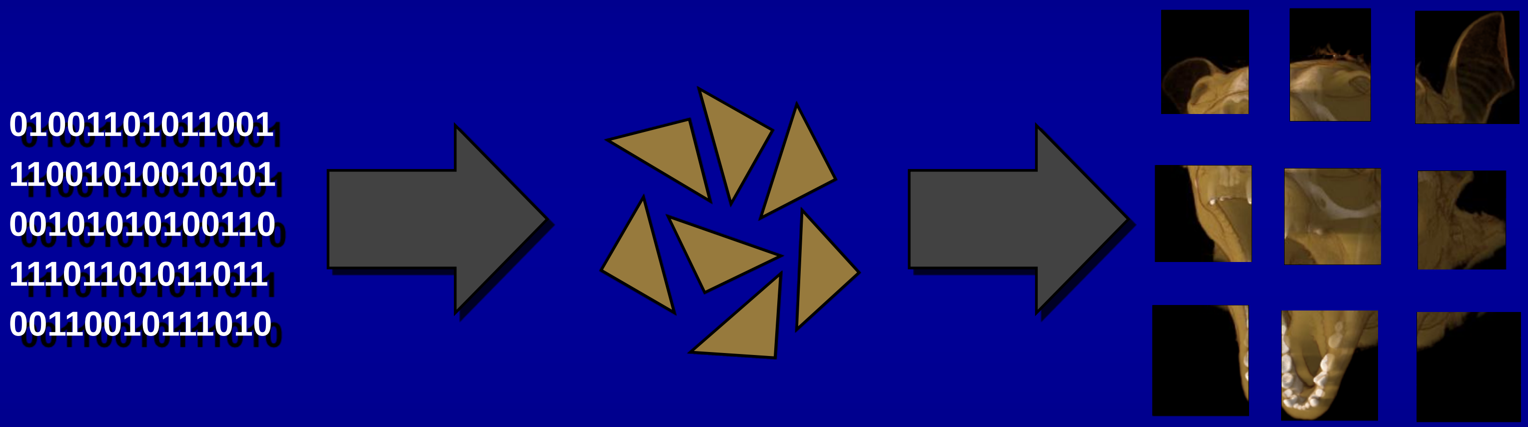
01001101011001 01001101011001
11001010010101 11001010010101
00101010100110 00101010100110
11101101011011 11101101011011
00110010111010 00110010111010

Or distributed rendering?



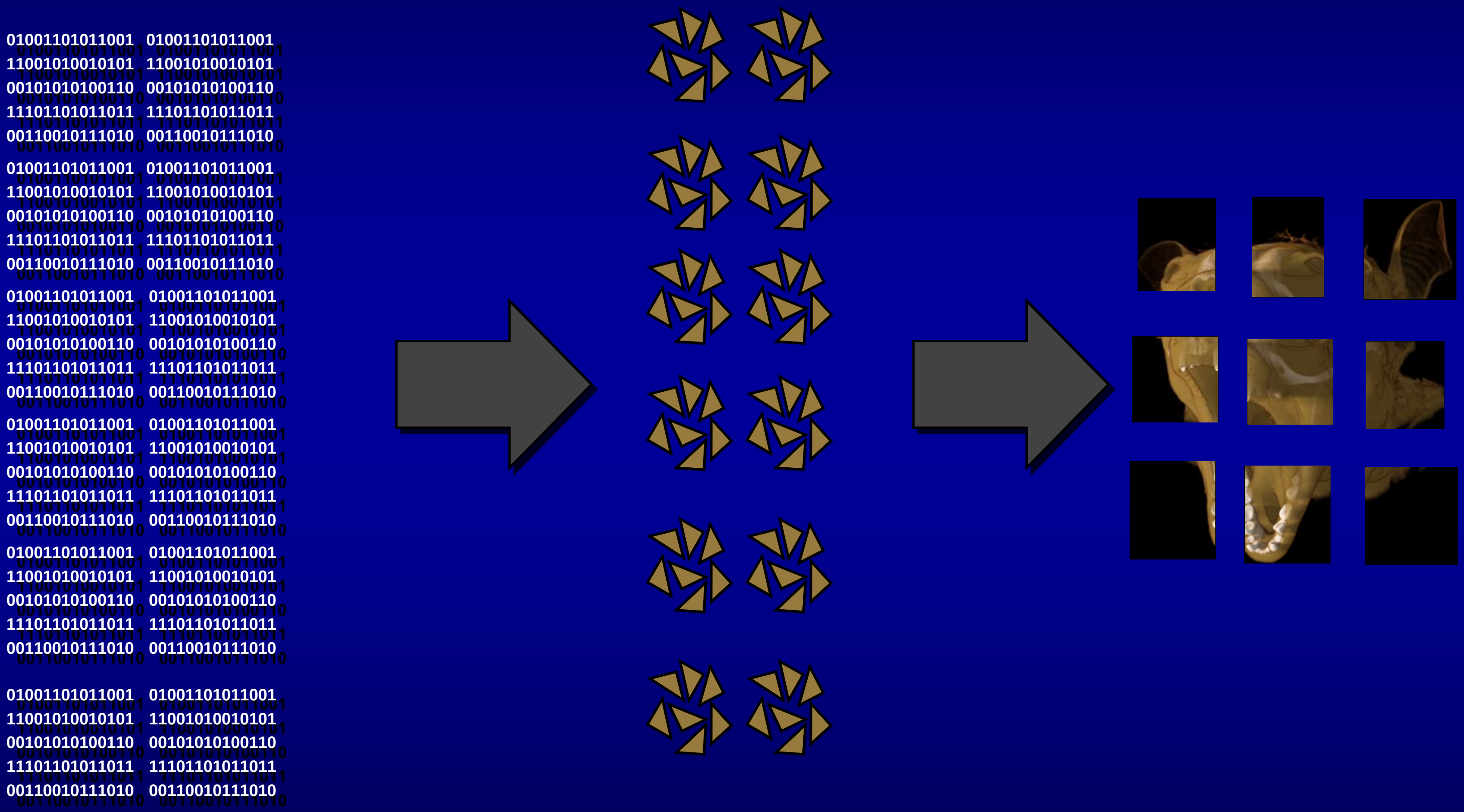
01001101011001
11001010010101
00101010100110
11101101011011
00110010111010

Or distributed displays?



01001101011001
11001010010101
00101010100110
11101101011011
00110010111010

Or all three?


01001101011001 01001101011001
11001010010101 11001010010101
00101010100110 00101010100110
11101101011011 11101101011011
00110010111010 00110010111010
01001101011001 01001101011001
11001010010101 11001010010101
00101010100110 00101010100110
11101101011011 11101101011011
00110010111010 00110010111010
01001101011001 01001101011001
11001010010101 11001010010101
00101010100110 00101010100110
11101101011011 11101101011011
00110010111010 00110010111010
01001101011001 01001101011001
11001010010101 11001010010101
00101010100110 00101010100110
11101101011011 11101101011011
00110010111010 00110010111010
01001101011001 01001101011001
11001010010101 11001010010101
00101010100110 00101010100110
11101101011011 11101101011011
00110010111010 00110010111010
01001101011001 01001101011001
11001010010101 11001010010101
00101010100110 00101010100110
11101101011011 11101101011011
00110010111010 00110010111010

Visualization Scaling Challenges

- Moving data to the visualization machine
- Most applications built for shared memory machines, not distributed clusters
- Image resolution limits in some software cannot capture feature details
- Displays cannot show entire high-resolution images at their native resolution

Visualization scales with HPC

Large data produced by large simulations require large visualization machines and produce large visualization results

Terabytes of Data

AT LEAST **Terabytes of Vis**

Gigapixel Images

Resampling, Application, …

Resolution to Capture Feature Detail

Moving Data

- How much time do you have?

File Size	**10 Gbps**	**54 Mbps**
1 GB	1 sec	2.5 min
1 TB	~17 min	~43 hours
1 PB	~12 days	~5 years

Analyzing Data

- Visualization programs only beginning to efficiently handle ultrascale data
  - 650 GB dataset -> 3 TB memory footprint
  - Allocate HPC nodes for RAM not cores
  - N-1 idle processors per node!
- Stability across many distributed nodes
  - Rendering clusters typically number N <= 64
  - Data must be dividable onto N cores *Remember this when resampling!*

Solution by Partial Sums

- Moving data – integrate vis machine into simulation machine. Move the machine to data!
  - Ranger + Spur: shared file system and interconnect

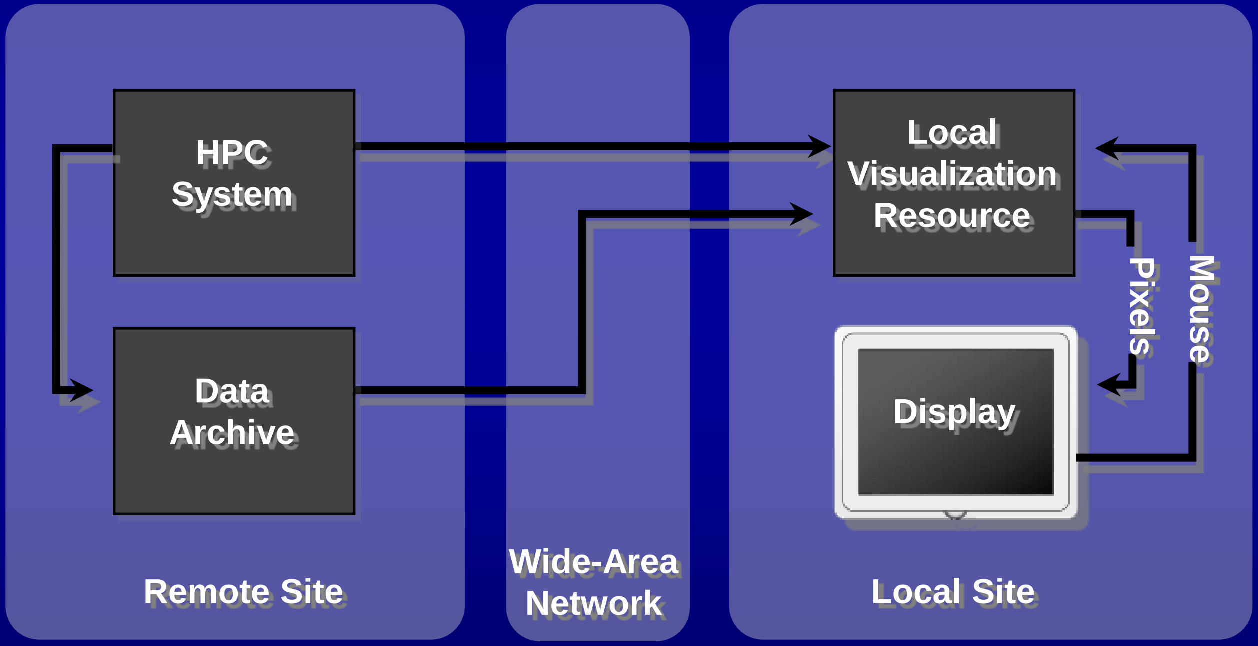
- Analyzing data – create larger vis machines and develop more efficient vis apps
  - Smaller memory footprint
  - More stable across many distributed nodes

Until then, the simulation machine is the vis machine!

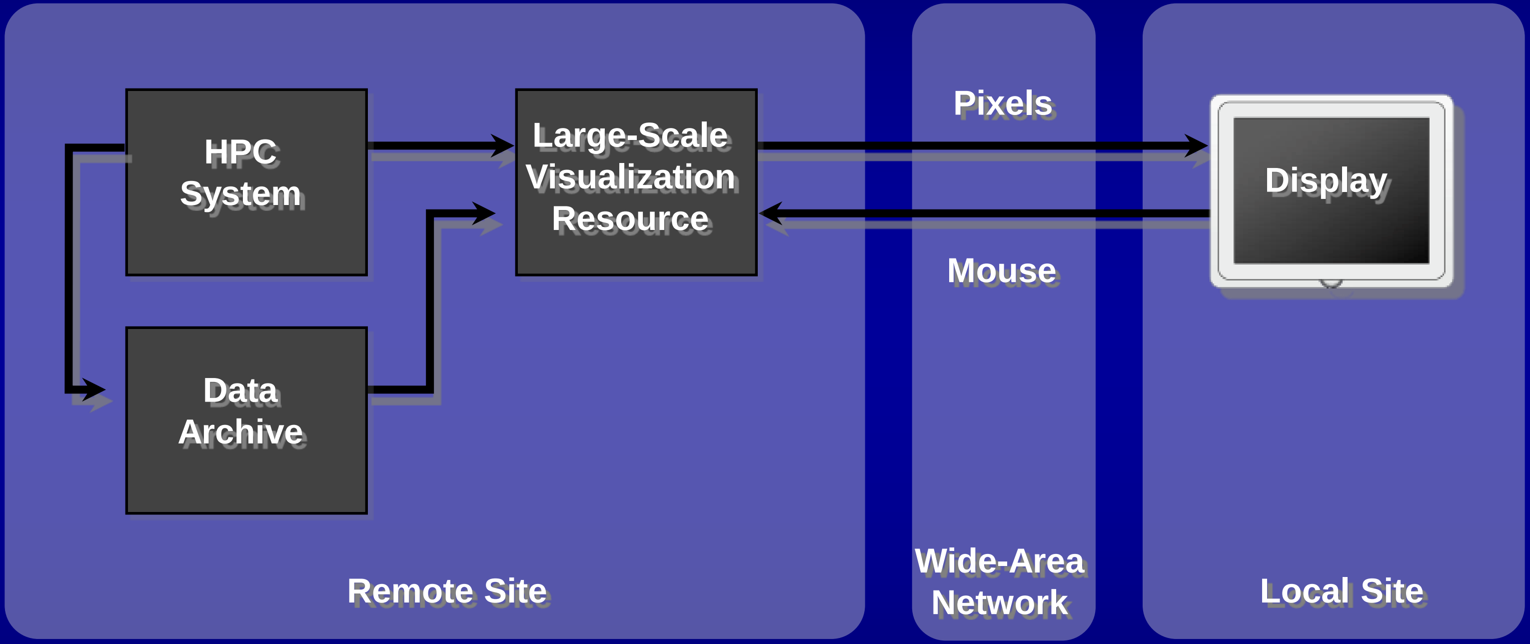
Solution by Partial Sums

- Imaging data – focus vis effort on interesting features
  parallelize image creation
  - Feature detection to determine visualization targets
    but can miss “unknown unknowns”
  - Distribute image rendering across cluster
- Displaying data – high resolution displays
  multi-resolution image navigation
  - Large displays need large spaces
  - Physical navigation of display provides better insights

Old Model
(No Remote Capability)

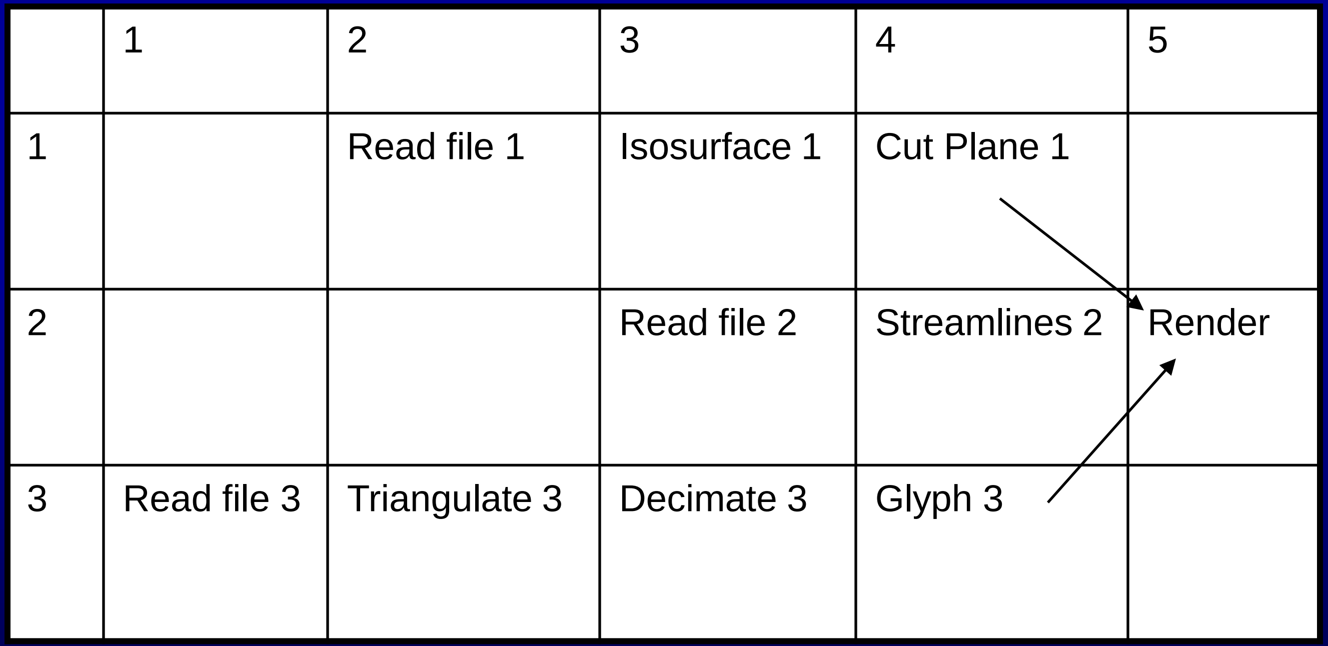


New Model
Remote Capability



Parallel Visualization

- Task parallelism – passing results to 1 process for rendering



Timesteps

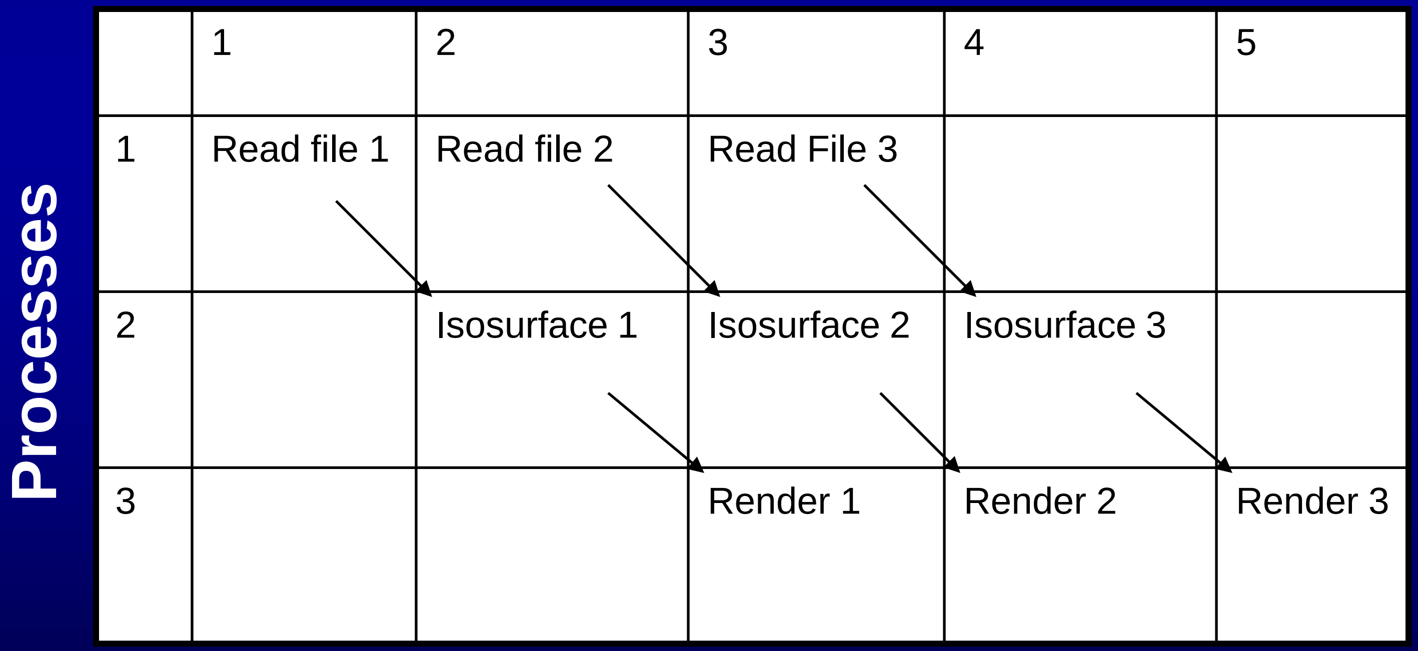
Processes

	1	2	3	4	5
1		Read file 1	Isosurface 1	Cut Plane 1	
2			Read file 2	Streamlines 2	Render
3	Read file 3	Triangulate 3	Decimate 3	Glyph 3	

Parallel Visualization

Pipeline parallelism

Useful when processes have access to separate resources or when an operation requires many steps.



Timesteps

Processes

	1	2	3	4	5
1	Read file 1	Read file 2	Read File 3		
2		Isosurface 1	Isosurface 2	Isosurface 3	
3			Render 1	Render 2	Render 3

Parallel Visualization

Data parallelism

Data set is partitioned among the processes and all processes execute same operations on the data.

Scales well as long as the data and operations can be decomposed.

Timesteps

Processes

	1	2	3
1	Read partition 1	Isosurface partition 1	Render partition 1
2	Read partition 2	Isosurface partition 2	Render partition 2
3	Read partition 3	Isosurface partition 2	Render partition 3

Parallel Visualization Libraries

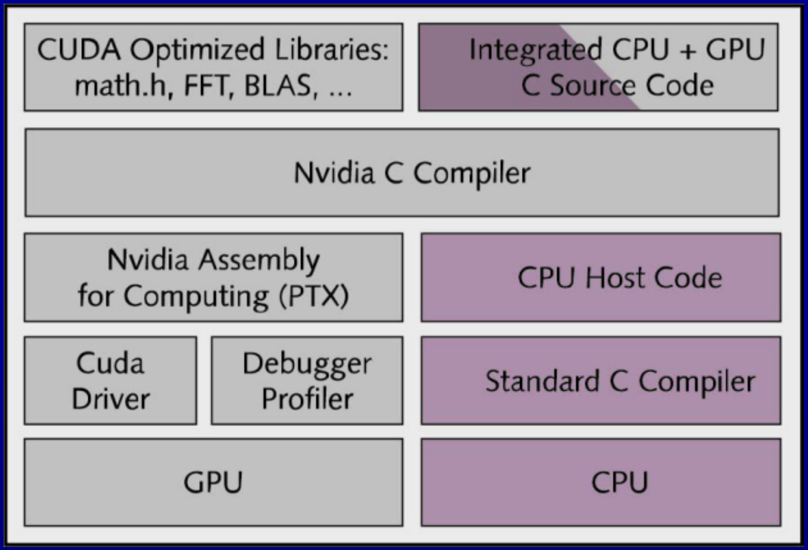
- Chromium – http://chromium.sourceforge.net
  - Sits between application and native OpenGL
  - Intercepts OpenGL calls, distribute across cluster
  - Can do either sort-first or sort-last
    (sort-first is simpler, sort-last can be better for large data)
  - Last update 31 Aug 2006, no new GL goodies
- IceT – http://www.cs.unm.edu/~kmorel/IceT/
  SAGE – http://www.evl.uic.edu/cavern/sage/
  CGLX – http://vis.ucsd.edu/~cglx/
  - specifically for large tiled displays
  - Must use IceT / SAGE / CGLX API in code
- Mesa – http://www.mesa3d.org/
  - Software rendering library
  - Enables OpenGL rendering on machines without GPUs

Open-Source Parallel Vis Apps

- VisIt – https://wci.llnl.gov/codes/visit/
  - Good scaling to hundreds of cores
  - Integrated job launching mechanism for rendering engines
  - Good documentation and user community
  - GUI not as polished
- ParaView – http://www.paraview.org/
  - Polished GUI, easier to navigate
  - Less stable across hundreds of cores
  - Official documentation must be purchased, though rich knowledge base on web (via Google)

CUDA – coding for GPUs

- C / C++ interface plus GPU-based extensions
- Can use both for accelerating visualization operations and for general-purpose computing (GPGPU)
- Special GPU libraries for math, FFT, BLAS



CUDA Optimized Libraries: math.h, FFT, BLAS, ...

Integrated CPU + GPU C Source Code

Nvidia C Compiler

Nvidia Assembly for Computing (PTX)

CPU Host Code

Cuda Driver

Debugger Profiler

Standard C Compiler

GPU

CPU

Image: Tom Halfhill, Microprocessor Report

GPU layout

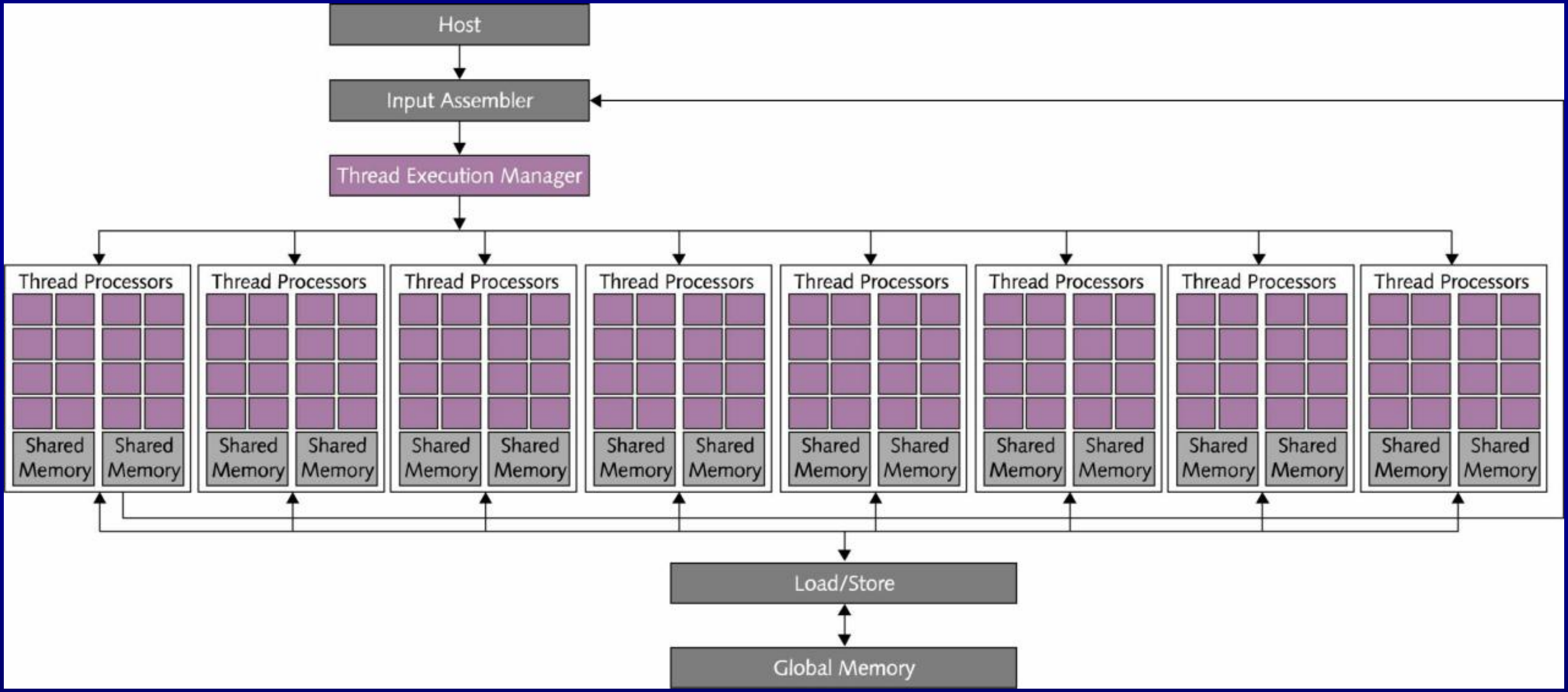


Image: Tom Halfhill, Microprocessor Report

GPU Considerations

- Parallelism – kernel should be highly SIMD
  - Switching kernels is expensive!
- Job size – high workload per thread
  - amortize thread initialization and memory transfer costs
- Memory footprint – task must decompose well
  - local store per GPU core is low (16 KB on G80)
  - card-local RAM is limited (~1GB on G8x)
  - access to system RAM is slow (treat like disk access)
- GPU N-body study (in GPU Gems 3): http://www.nvidia.com/object/io_1195170003876.html

Summary

- Challenges at every stage of visualization when operating on large data
- Partial solutions exist, though not integrated
- Problem sizes continue to grow at every stage
- Vis software community must keep pace with hardware innovations

Thank you!

kelly@tacc.utexas.edu