Visualization on JURECA
Outline

- Remote Rendering
- In-Situ Visualization
Visualization: General Hardware Setup

12 Visualization Nodes
- 2 GPUs Nvidia Tesla K40 per node
  - 12 GB RAM on each card

- 2 Login Visualization Nodes
  - jurecavis.fz-juelich.de

- 10 Batch Visualization Nodes
  - 8 nodes with 512 GB RAM
  - 2 nodes with 1024 GB RAM
  - Special partition named “vis”

Visualization also possible on nodes without GPU’s (software rendering)
Visualization: Remote Rendering

- **vis login node:**
  - direct user access
  - no accounting
  - shared with other users
  - no parallel jobs (no `srun`)

- **vis batch node:**
  - access via batch system
  - accounting
  - exclusive usage
  - parallel jobs possible
**Visualization: Remote Rendering**

**Visualization Scenarios:** using remote graphical desktop (with VNC)

Visualization Scenarios: non-VNC
client runs on users workstation and connects to a server on JURECA
Visualization: VNC – remote graphical desktop

- hardware rendering (GPU acceleration) with VirtualGL
- only (compressed) images are transferred
- interactive frame rates with moderate WAN bandwidth
- look-and-feel like a local desktop
- direct access to GPFS
- high network bandwidth, latency, quality of rendering

Attention:
- start any OpenGL application with “vglrun”
- make sure it appear on the “GPU utilization” panel

https://trac.version.fz-juelich.de/vis/wiki/vnc3d
Visualization: VNC – remote graphical desktop

Strudel (ScienTific Remote Desktop Launcher)
Complex VNC scenarios become easy to use for any user

1) Install TurboVNC
2) Load your SSH key into the SSH key agent
3) Start Strudel and login

Download & Install instructions:
https://trac.version.fz-juelich.de/vis/wiki/vnc3d/strudel
Outline

• Remote Rendering
• In-Situ Visualization
In-Situ Visualization: Motivation

- Visualizing highly resolved simulations can easily result in large amount of data.
- Reading/Writing this data can become far too expensive.
- In-Situ visualization can avoid this unnecessary reading/writing.
- Not only for visualization, but for post-processing in general I/O can become a major bottleneck. => in-situ processing
**In-Situ Visualization: Common Software**

**VisIt**
developed by multiple national US labs funded by Department of Energy (DOE) initial release 2002 hosted at LLNL open source (BSD)

**ParaView**
developed by Kitware + Los Alamos National Lab funded by Department of Energy (DOE) initial release 2002 hosted by Kitware open source (BSD)
In-Situ Visualization: with VisIt/Libsim

1. Launch simulation
2. Remote VisIt connects to simulation
3. Simulation becomes Engine
4. Engine pulls data
5. Engine processes+(renders) data and commands from GUI requests
6. View (renders)+displays data
In-Situ Visualization: with VisIt/Libsim

The simulation windows shows the meta-data about the running code.

Control commands defined by the simulation code accessible here.

User selects running simulations to connect to as if they were files.

All VisIt existing functionality is accessible.

June 7th, 2016
In-Situ Visualization: with VisIt/Libsim

https://trac.version.fz-juelich.de/vis/wiki/VisIt/libsim
In-Situ Visualization: Quick & Easy

Virtual-Box Images are available (on USB-stick) to have an easy start with In-Situ Visualization based on VisIt/libsim and JUSITU.

1) get the „JUSVis“ VirtualBox-Image
2) start VirtualBox and import appliance (File->Import Appliance)
   user: jscvis   passwd: jscvis   root-passwd: jscvis

3)  cd ~/Software/insitu/
4)  mkdir jusitu-build; cd jusitu-build
5)  cmake ..../jusitu-15042016; make
6)  cd examples/JUSSim; ./JUSSim

7)  start VisIt and open ~/visit/simulation/…simulation.sim2
8)  Add-> Mesh3D; Draw

VisIt/Libsim examples

JUSITU - https://gitlab.version.fz-juelich.de/vis/jusitu
Visualization with VisIt Tutorials

- Tutorial Preparation
- VisIt-tutorial-basics
- VisIt-tutorial-data-analysis
- VisIt-tutorial-Python-scripting

- Blood_Flow_Aneurysm_Tutorial_Dataset_Exploration
- Blood_Flow_Aneurysm_Tutorial_Vector_Field_Visualization
- Blood_Flow_Aneurysm_Tutorial_Calculating_Flux
JURECA Visualization Related Documentation

Entry point is

https://trac.version.fz-juelich.de/vis/

Docu related to VNC (remote graphical desktop)

https://trac.version.fz-juelich.de/vis/wiki/vnc3d

Docu related to VisIt:

https://trac.version.fz-juelich.de/vis/wiki/VisIt

Docu related to ParaView:

https://trac.version.fz-juelich.de/vis/wiki/ParaView

GitLab of JUSITU:

https://gitlab.version.fz-juelich.de/vis/jusitu
Thank you for your attention

Questions?

rendered with Blender from a DNS of a diesel injection spray of ITV, RWTH Aachen University