GPU Rendering in Rhea and Titan

Benjamin Hernandez

Advanced Data and Workflows Group
GPU Rendering in OLCF

• For traditional or custom applications that uses intensively the GPU for visualization, OLCF supports VirtualGL/TurboVNC.

• So if you have software or a custom visualization tool you can take advantage of this feature.

• ParaView implement their own mechanism for delivering frames to the user’s desktop through a client-server architecture.
GPU Rendering in OLCF

• Traditional X-forwarding.

```bash
ssh -X user@titan.ccs.ornl.gov
```

![Diagram](attachment:image.png)

• GPU based rendering is quite different to X forwarding
GPU Rendering in OLCF

• What about NoMachine’s NX?
  – It uses X compression
  – But do not take advantage of remote GPUs
VirtualGL/TurboVNC

Titan / Rhea

Application

libGL  Xlib

VirtualGL  vncserver

3D X Server

Driver

GPU

Your machine

Compressed X commands

Compressed Frames

vncviewer

Driver

GPU / Monitor
Prerequisites

• You’ll need to have turboVNC installed on your machine.
• TurboVNC uses 59XX ports for tunneling
• You’ll need to open 3 Terminals
Specs

Titan
- 16 600+ nodes
- 16 600+ K20, 6GB
- 32 GB RAM
- 16 CPU Cores, 2.2 GHz

Rhea (Analysis Cluster)
- 512 nodes
- 9 Fat nodes:
  - 2x NV K80, 24 GB
  - 1 TB RAM
  - 2x CPU 28 Cores 2.3GHz
GPU Rendering on Rhea Terminal 1:

- Login to Rhea using login 5 node

  `ssh user@rhea-login5.ccs.ornl.gov`

- Submit an interactive job in the GPU partition

  `qsub -I -lnodes=1,walltime=01:00:00,partition=gpu -A abc123`

- Launch x-server

  `xinit&`

- Launch the vncserver

  `vncserver :1 -geometry 1900x1000 -depth 24`

- Check where is running the vnc server, e.g.

  New ‘rhea-gpu3:1 (user)’ desktop is `rhea-gpu3`

- Type

  `export DISPLAY=:1`
GPU Rendering on Rhea

Terminal 2:

- We will open a tunneling connection to the node running vncserver, in this case `rhea-gpu3` through `rhea-login5`
  
  ```
  ssh user@rhea-login5.ccs.ornl.gov -L 5901:rhea-gpu3:5901
  ```

Terminal 3:

- In your machine run VNC viewer using the tunneling port
  
  ```
  vncviewer -medqual localhost:5901
  ```
GPU Rendering on Titan
Terminal 1:

- Login to Titan and allocate compute resources through the batch system
  
  \texttt{qsub -I -lnodes=1 -lwalltime=01:00:00 -Aabc123}

- From within the batch job, set-up the environment
  
  \texttt{module load GPU-render}

- After loading the module, you’ll see the node name where is running your job, e.g.

  \texttt{nid02410}

- From within the batch job, run a script on the compute nodes that starts X and runs a test.

  \texttt{aprun -n $PBS\_NUM\_NODES $MEMBERWORK/abc123/job.sh}
GPU Rendering on Titan Interactive Job

#!/bin/sh
startx &
sleep 5
starttvtnc :1 &
export DISPLAY=:1
vglrun -np 4 ./yourapp
GPU Rendering on Titan

Terminal 2:
- We will open a tunneling connection to the node running vncserver, in this case **nid02410** through **titan-internal**
  
  `ssh user@titan-internal.ccs.ornl.gov -L 5901:nid02410:5901`

Terminal 3:
- In your machine run VNC viewer using the tunneling port
  
  `vncviewer -medqual localhost:5901`
More Info. and help

https://www.olcf.ornl.gov/kb_articles/gpu-rendering-on-titan/

help@olcf.ornl.gov

Thanks!