

Remote Rendering

Das Remote-Rendering per EnginFrame wird zur Zeit nicht unterstützt, die entsprechenden Nutzungslizenzen sind ausgelaufen und das Paket wird zeitnah durch den Dienst "PBS-Access" ersetzt. X11-forwarding (s.u.) funktioniert weiterhin.

~~Es ist möglich eine Remote-Rendering-Sitzung innerhalb des Clusters zu nutzen um gui-basierte und grafiklastige Programme auszuführen. Dazu wird die Software EnginFrame verwendet. Das Rendering findet ganz normal auf den Knoten statt und es kann ausgesucht werden, ob eine GPU benötigt wird. Die meisten Programme funktionieren ohne Leistungseinbußen obwohl keine GPU genutzt wird.~~

~~Schritt für Schritt-Anleitung~~

1. Melden Sie sich bei EnginFrame an: <https://view-2018.hpc.rz.uni-duesseldorf.de/enginframe/vdi>
2. Klicken Sie Links auf "Linux-Desktop" unter Session
3. Tragen Sie ihre myJAM-Projekt und die Ressourcen-Anforderungen ein
4. Klicken Sie auf Launch Session
5. Je nach Auslastung des Clusters sollte nach einigen Sekunden ein neuer Browser-Tab geöffnet werden mit der Remote-Session

Alternative: X11-Forwarding

Wenn eine grafische Benutzeroberfläche gewünscht wird, kann alternativ zur obigen EnginFrame-Lösung ein X11-Forwarding genutzt werden. Dies geht in zwei Schritten:

1) Linux unterstützt X11 nativ - dort geschieht dies mit der SSH-Option "-X", d. h. man loggt sich mittels "**ssh -X <Nutzerkürzel>@[hpc.rz.uni-duesseldorf.de](https://view-2018.hpc.rz.uni-duesseldorf.de)**" auf dem Login-Knoten ein.

Windows unterstützt X11 leider (noch) nicht nativ - dort benötigt man extra Software wie MobaXterm, wo man in den Einstellungen (Advanced SSH Settings) die Option "X11-Forwarding" aktivieren muss.

Um zu schauen, ob dieser erste Schritt geklappt hat, kann man auf die Ausgabe von "**echo \$DISPLAY**" schauen - wird da etwas (wie z. B. "localhost: 11.0") angezeigt, dann funktioniert das erste X11-Forwarding.

2) Danach startet man einen interaktiven Job mittels qsub, allerdings auch mit der gleichen Zusatzoption "-X", also z. B. "**qsub -l -X job_resources.sh**".

Auch hier sollte die Display-Variable gesetzt sein. Nun kann man Anwendungen auch mit einer grafischen Oberfläche starten.

Beispiel mit Matlab:

```
$ module load MATLAB/R2018a
```

```
$ matlab
```

```
MATLAB is selecting SOFTWARE OPENGL rendering.
```

```
...
```

(Matlab erkennt also automatisch, dass eine grafische Ausgabe möglich ist und nutzt diese dann.)