Integrating VR SLI into Autodesk VRED

Michael Nikelsky
Principal Engineer
We may make statements regarding planned or future development efforts for our existing or new products and services. These statements are not intended to be a promise or guarantee of future availability of products, services or features but merely reflect our current plans and based on factors currently known to us. These planned and future development efforts may change without notice. Purchasing decisions should not be made based upon reliance on these statements.

These statements are being made as of April, 4th 2016 and we assume no obligation to update these forward-looking statements to reflect events that occur or circumstances that exist or change after the date on which they were made. If this presentation is reviewed after April, 4th 2016, these statements may no longer contain current or accurate information.
Autodesk VRED Professional

- Visualization and virtual prototyping tool
  - Focus on Automotive
  - Powerful data preparation and analysing tools
  - High quality OpenGL and raytracing rendering
  - VR support
    - Powerwalls, Cave
    - Oculus Rift
    - HTC Vive
Why we need VR SLI

- VR headsets are becoming popular again
  - Virtual Seating Booth
  - Virtual Factory
  - Driving Simulations
  - Design Reviews
  - Sales and Marketing
Why we need VR SLI in VRED

- Engineering Datasets
  - 30-60M triangles inside the view frustum
  - ~3-5k Geometries
  - 100-300 materials
- Complex shaders
  - Realistic appearance
  - Measured materials
- No extra data preparation step possible
Stereo Rendering in VRED

- Renderengine based on OpenGL 4.x
  - Uniform buffer required for camera data
- Standard Stereo Rendering
  - For each eye
    - Bind Rendersurface
    - Setup Camera Buffer
    - Render the scene
  - Submit rendersurfaces to HMD
VR SLI Rendering in VRED

- Init the multicast SLI mode
  - GL_NVX_LINKED_GPU_MULTICAST=1
- Modified renderloop
  - Bind rendersurface
  - Setup Camera Buffer for both eyes
  - Render the scene
  - Copy rendersurface from GPU1 to GPU0
  - Submit rendersurfaces to HMD
Nasty little details

- **View Frustum Culling**
  - Different objects for left and right eye
  - Only one eye is rendered
  - Need to adjust view frustum to enclose both eyes
- **Occlusion queries do not work**
  - GPU side occlusion culling might work
Hidden Area Mesh rendering is tricky
  - Upload different mesh data to the different GPUs
  - Flip the left eye mesh for the right eye
  - Skip it if you use timewarping

Careful with Multi-CPU Systems
  - Both GPUs need to be attached to one CPU
Results

- **Stereo Rendering in FullHD**
  - Up to 88% better Performance compared to single GPU Stereo rendering
  - About 95% of the performance compared to mono rendering
  - CPU load is cut in half

<table>
<thead>
<tr>
<th></th>
<th>No AA</th>
<th>Low AA</th>
<th>High AA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mono</td>
<td>53</td>
<td>45</td>
<td>33</td>
</tr>
<tr>
<td>Single GPU Stereo</td>
<td>27</td>
<td>23</td>
<td>17</td>
</tr>
<tr>
<td>VR SLI</td>
<td>50</td>
<td>42</td>
<td>32</td>
</tr>
</tbody>
</table>
## Results

- **HMD Rendering**
  - Vsnc/Syncpoints make accurate measurement difficult
  - Performance improvement ranges from 40% to 100% depending on Vsnc/Syncpoint miss

<table>
<thead>
<tr>
<th></th>
<th>No AA</th>
<th>Low AA</th>
<th>High AA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single GPU Stereo</td>
<td>56</td>
<td>45</td>
<td>45</td>
</tr>
<tr>
<td>VR SLI</td>
<td>90</td>
<td>90</td>
<td>62</td>
</tr>
</tbody>
</table>

### VR SLI Scaling

![VR SLI Scaling Graph]

- **No AA**: 101
- **Low AA**: 81
- **High AA**: 61