Next-Generation GPU Rendering: High-End Production Features on GPU

Blagovest Taskov
Lead Developer
V-Ray GPU
GPGPU in Chaos Group

- V-Ray & **V-Ray (RT) GPU**
- CUDA & OpenCL
- **Denoiser**
- VFB effects
Mastering CGI
Model by Darren Wallace
Pascal & NVLINK

- Makes it possible to access data between GPUs
  - So 2 GPUs, with 16GB memory each results in 32GB memory
- NVLINK supported completely in V-Ray GPU for textures & geometry
- Round-robin strategy seems to work perfectly
- Unexpectedly good performance
- 1st out of NVIDIA implementation
NVIDIA MDL

- GLSL textures first supported in V-Ray GPU
- MDL is far superior to OSL and GLSL for GPU rendering
- Supported both in V-Ray and V-Ray GPU as of version 3.5
- Practically allowing rendering of any kind of texture or BRDF
- Substance designer, IRay, MentalRay, Adobe
GPU Memory Management

- The traditional “out-of-core” architecture has a lot **drawbacks**
  - Requires substantial GPU memory buffers to keep the GPU busy
  - Gets really slow when the real “out-of-core” kicks in
- **On-demand mip-mapping**
  - Create mip—maps on demand from any texture format
  - Detect automatically how many channels the textures need, no extra work required from artist
- **No GPU memory used for frame buffer**
  - No problem to render 50k image with 20 render elements on a GTX 970
Adaptive lights
Adaptive lights

- New unbiased path-tracing algorithm designed from scratch for GPUs. Works with BF/LC/IM. No need of extra large memory buffers.
- Works by examining the scene runtime and changing sampling techniques (kernels) based on the scene.
- No need of adjusting “light samples”, nor turning MIS on/off
- Allows the GPU rendering to scale into the future
• Speeds up scenes with 1 light as well.
• Works with progressive rendering (no need for complex prepass)
• Allows the GPU rendering to scale into the future
• Designed for and working great on GPU
Better materials sampling in Maya
Stochastic Flakes BRDF

- Based on paper that Chaos Group presented at SIGGRAPH 2016
- Mathematically modeled BRDF that simulates mirror-like flakes
- Algorithm designed to work on GPU
Rect Light Directionality
Aerial perspective
Environment Fog *(in nightlies)*

*Fluids*
Forest Color

- First Forest Color implementation in a GPU renderer
- Exact the same look as on a CPU
  - You can’t get the same result with regular “scatter/random”
- Engineered with the help of ITooSoftware
- Very quick to render
V-Ray Denoiser

- Works both with CPU and GPU renders
- GPU Accelerated. More than x25 speedup compared to CPU.
- Does not mandate OpenCL devices
- Interactive, non-destructive denoising during render time
- Uses data from multiple frames to denoise animations.
- Available as free plugin for Nuke as well
- Is practically impossible with CPU based algorithms
One more thing ...
• Many peoples happen to have quite a lot of CPU power, but want to switch to GPU rendering.
• CPU often has more memory.
• Every machine happens to have a CPU. Most of the time during GPU rendering, the CPU does nothing.
• There are still many CPU render farms.
V-Ray GPU hybrid rendering

• You can use **CPU** just like regular GPU CUDA device when rendering with V-Ray GPU CUDA.

• The CPU executes exactly the same code, so it **renders exactly the same**. You can mix CPU and GPU frames.

• Makes switching to GPU for people with existing CPU farms much easier.

• **Using all the available hardware** is always a good idea and makes the rendering faster.

• Available in nightlies.
DEMO
Thanks
And may the flops be with you

chaosgroup.com
blagovest.taskov@chaosgroup.com
facebook.com/groups/VRayRT