RTX-accelerated Hair brought to Life with NVIDIA Iray (GTC 2020 S22494)

Carsten Waechter, March 2020
What is Iray?

Production Rendering on CUDA

Bring ray tracing based production / simulation quality rendering to GPUs

New paradigm: *Push Button* rendering (open up new markets)

Plugins for

3ds Max  Maya  Rhino  SketchUp  ...

In Production since > 10 Years

SUBSTANCE DESIGNER  SUBSTANCE PAINTER

SIEMENS NX  SOLIDWORKS  CATIA

Daz 3D  ingenius  amazon  ...

...
What is Iray?

**NVIDIA testbed and inspiration for new tech**

NVIDIA Material Definition Language (MDL)
- evolved from internal material representation into public SDK

NVIDIA OptiX 7
- co-development, verification and guinea pig

NVIDIA RTX / RT Cores
- scene- and ray-dumps to drive hardware requirements

NVIDIA Maxwell…NVIDIA Turing (& future) enhancements
- profiling/experiments resulting in new features/improvements

Design and test/verify NVIDIA’s new Headquarter (in VR)
- close cooperation with Gensler
Artistic Freedom
How Does it Work?

99% physically based Path Tracing

To guarantee simulation quality and *Push Button*

- Limit shortcuts and good enough hacks to minimum
- Brute force (spectral) simulation
  - no intermediate filtering
  - scale over multiple GPUs and hosts even in interactive use
- Two-way path tracing from camera and (opt.) lights
- Use NVIDIA Material Definition Language (MDL)
- NVIDIA AI Denoiser to clean up remaining noise
How Does it Work?

99% physically based Path Tracing

To guarantee simulation quality and **Push Button**

- Limit shortcuts and good enough hacks to minimum
- Brute force (spectral) simulation
  - no intermediate filtering
  - scale over multiple GPUs and hosts even in interactive use
- Two-way path tracing from camera and (opt.) lights
- Use NVIDIA Material Definition Language (MDL)
- NVIDIA AI Denoiser to clean up remaining noise
Wavefront Architecture

From Megakernel to State Machine

Follows each path to completion
One path at a time
Single CUDA (mega-)kernel

Small progress on each path per step
Millions of active paths at a time
Multiple smaller CUDA kernels (states) specialized on parts of the simulation (state machine)

Global memory (AoSoA layout) to communicate between states
Iray on OptiX 7

Wavefront Architecture

All kernel variants that need to trace rays are now executed through OptiX 7

Path-/Light-Tracer main trace kernels
incl. SSS code and shortcuts for state machine early outs
Iray on OptiX 7
Wavefront Architecture

All kernel variants that need to trace rays are now executed through OptiX 7

Path-/Light-Tracer main trace kernels  
  incl. SSS code and shortcuts for state machine early outs

Path-/Light-Tracer shadow trace kernels  
  incl. few shortcuts for state machine early outs

Rounded Corners

Light-Tracer lens connection

All other kernels stay on plain CUDA implementations / kernel launches (for now)
Iray on OptiX 7
Wavefront Architecture

Split up the Tail-megakernel into 2 new kernels
Trace rays + the *remainder* of the state machine

**Majority of code in **__**raygen**__**
One single `optixTrace()` call, no branching, for best performance
(except for Tail-trace- and rounded corners kernels)

**__closesthit__** directly fills wavefront state, no payload communication

Compile time / Pipeline setup 7-10 secs (with warm cache 0.1-0.2 secs)

~21k lines of PTX
New in 2020.0: Curves / Fibers
How Does it Work?

Coop development on new OptiX 7.1 curve API

Iray 2020.0 exposes a subset

- **Cubic B-Spline Basis**
  - With vertex sharing (saves memory & bandwidth)
  - X curves combined into 1 connected fiber

- **ISV responsible for conversion from spline bases to B-spline**
  - Memory cost: no vertex sharing
  - Bezier and anything compatible, e.g., Catmull-Rom, Hermite, ...

- **Intersection code based on (improved) NVIDIA research tech**
  - Fast, High Precision Ray/Fiber Intersection using Tight, Disjoint Bounding Volumes *Nikolaus Binder and Alexander Keller*
How Does it Work?

Fiber rendering

Material and Texture inputs

- MDL 1.6 hair BSDF
  A Practical and Controllable Hair and Fur Model for Production Path Tracing Chiang et al.

- Texture space
  0: 1D along fiber [0..1]
  1: per fiber: either user provided or (by default) origin position of fiber in world space (1D, 2D or 3D)
  2: per vertex: user provided (1D, 2D or 3D)
How Does it Work?
Fiber rendering

Intersection

• Separate hierarchies for triangles and fibers
• First trace triangle scene, then fibers for efficiency
• When using MDL hair BSDF
  “Teleport” intersection point to other side of the fiber, along normal, to be used as exit point
  BSDF is supposed to handle most internal effects
• Continue with self intersection handling code

A Fast and Robust Method for Avoiding Self-Intersection
Carsten Waechter and Nikolaus Binder
When Does it Not Work?
Fiber rendering

Internal rays
• Current implementation limitation: Rays starting inside a fiber will lead to undefined results, as considered solid
• Thus: Secondary rays from fiber hits should be launched from outside any fibers, which is difficult to detect (e.g. millions of hairs)
• This limitation will hopefully vanish soon (newer OptiX 7 releases)
• Artifacts usually (e.g. millions of hairs) not visible though
How Fast is it?

Benchmark

Absolute: < 1min beauty FullHD

> 6 million fibers + MDL hair BSDF

Benchmarking different generations

• Exceptional performance increase
  Comparing RTX on vs off

• And even when comparing exceptional triangle scenes

• So (usually) no need to triangulate for performance
Questions?

Acknowledgments

Iray Team / NVIDIA ARC Berlin

More Information

Techreport: The Iray Light Transport Simulation and Rendering System

Other sessions featuring Iray

Alita, Substance, and RTX [S22395]
David Crabtree, Build Lead, DNEG

Visuals as a Service (VaaS):
How Amazon and Others Create and Use Photoreal On-Demand Product Visuals with RTX Real-Time Raytracing and the Cloud [S21290]
Paul Arden, CEO, migeniuss
Thomas Dideriksen, Senior Software Developer, Amazon

Sharing Physically Based Materials Between Renderers with MDL [S21220]
Lutz Kettner, Director, Adv. Rendering and Materials, NVIDIA
Jan Jordan, Senior Software Product Manager, NVIDIA

Photoreal Design Workflows with NVIDIA Iray: the Siemens Experience [S22454]
Patti Longwinter, Senior Product Manager, Siemens
Alexander Fuchs, Senior Software Product Manager, NVIDIA