GI NEXT

GLOBAL ILLUMINATION FOR PRODUCTION RENDERING ON GPUS

May 10th, 2017

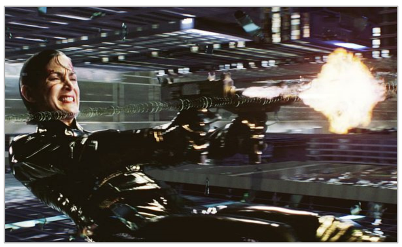
NVIDIA

Enzo Catalano, Rajko Yasui-Schoeffel

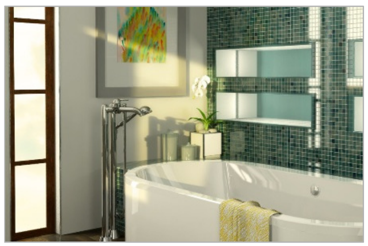
Senior Graphics Software Engineers, mental ray®

PRODUCTION RENDERING

How to get the best of both CPU and GPU in a production renderer



From "The Matrix Reloaded". ©Warner Brothers, Village Roadshow Pictures. Digital work by ESC Entertainment.



Rendering using ViSoft® Photo Tuning in ViSoft® Premium, ViSoft® GmbH, Germany.

Rollin Wild, ©2012 Filmakademie Baden-Württenberg

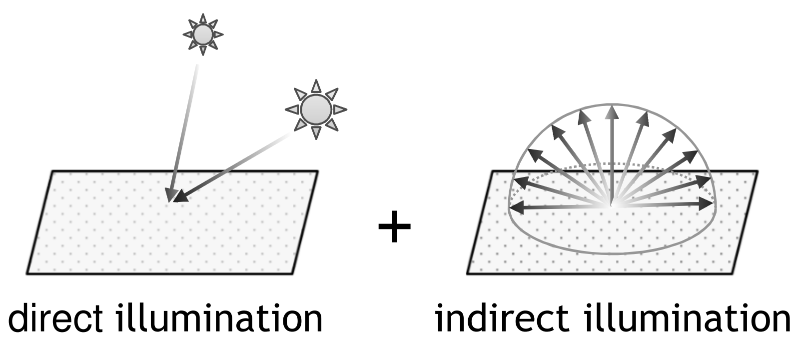
From "Star Wars: Episode II - Attack of the Clones". Lucasfilm Ltd.™. Digital work by ILM.

From "Brothers Grimm". Dimension Films. Digital work by Peerless Camera Company, Ltd.

- ray tracer calling shaders calling core functionality
  - massive assets: large geometry, hair, fur, foliage, huge textures and many lights
  - global illumination with motion blur, depth of field…
- shaders may be custom written, even black boxes

PRODUCTION RENDERING

Global illumination



Example: NVIDIA® mental ray® **`mi_compute_avg_radiance`** core function

indirect (diffuse) illumination by shooting rays into the hemisphere

EVALUATION OF INDIRECT ILLUMINATION

Issues with previous approaches

- interpolation of cached samples suffers from notorious artifacts
  - requires time consuming tuning and repeated renders
  - tends to have many controls
- brute force methods typically very slow
  - simpler to control, delivering desired high quality
  - many rays to be traced and shaders to be called at each hit point
- still, may get away with brute force and approximate samples

GI NEXT

Accelerated indirect illumination by approximate shading

- path tracing on the original geometry
- using one generic Bidirectional Scattering Distribution Function (BSDF)
  - BSDF parameters stored in a point cloud
    - extracted from the original shaders
    - in addition, local surface properties and direct light contribution are stored
- feed the BSDF with parameters from the nearest neighbor in the point cloud
  - saves shader calls, especially light loops and texture accesses

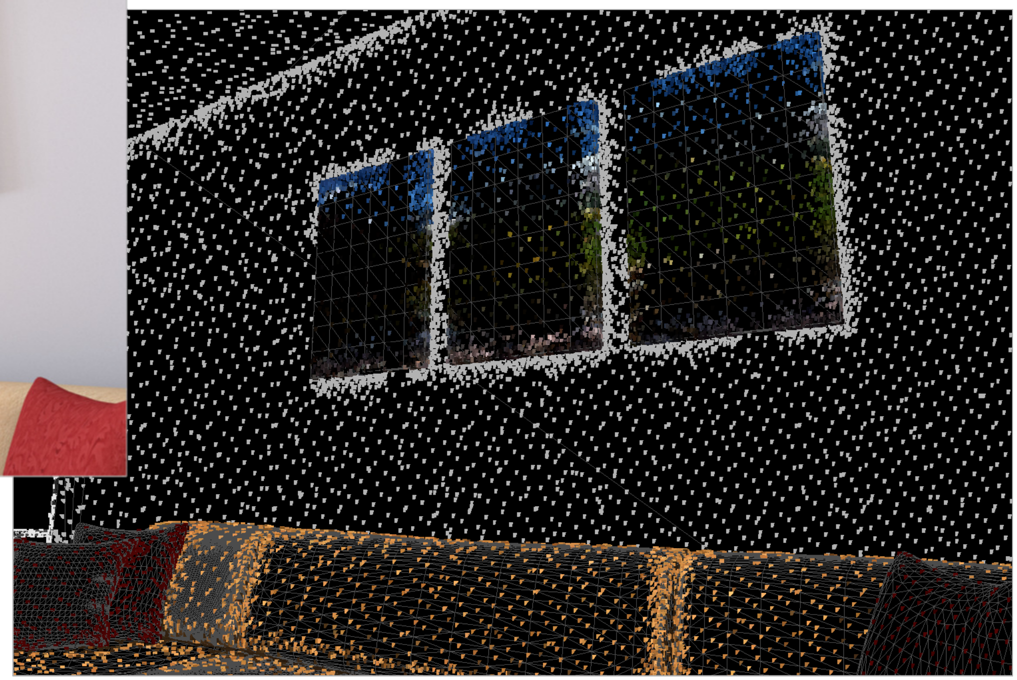
GI NEXT

Creating point clouds

- sample points on objects in the scene
  - infer BSDF parameters by shader exploration, which also samples texture information
  - compute and store direct illumination
  - adaptive sampling organized in passes

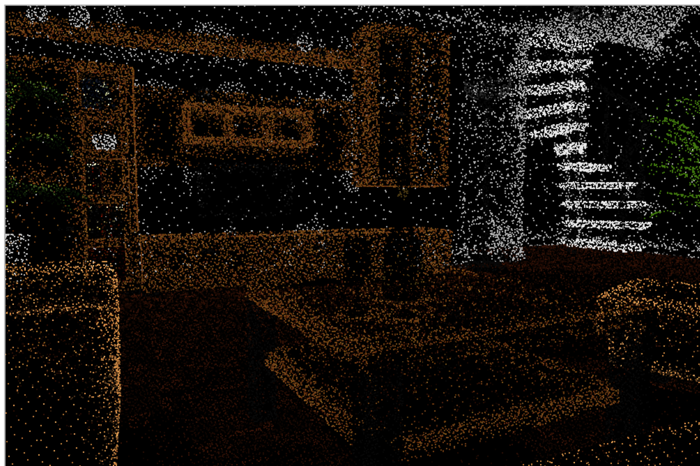
GI NEXT

Creating point clouds

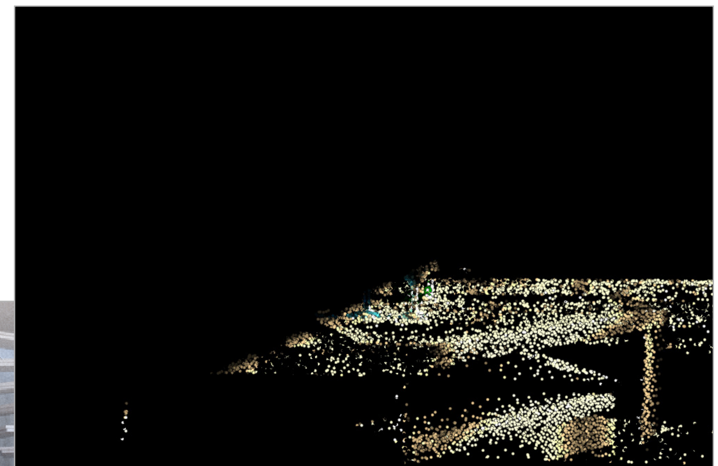


GI NEXT

Creating point clouds



diffuse color



direct illumination

GI NEXT

Shader exploration: The principle by example

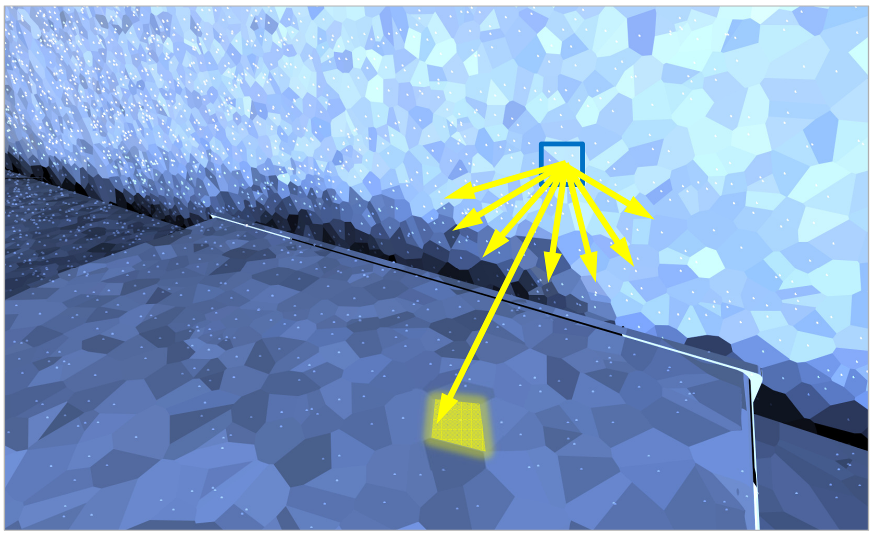
determining the diffuse albedo of the BSDF assuming that

- color = `diffuse * ( light_loop() + mi_compute_avg_radiance() ) + ambient`
- first shader invocation: core to turn off all illumination
  - yields `c1 = ambient`
- second shader invocation: core to return white for `mi_compute_avg_radiance()`
  - yields `c2 = diffuse + ambient`
- hence, the diffuse albedo of the BSDF is `c2 - c1`

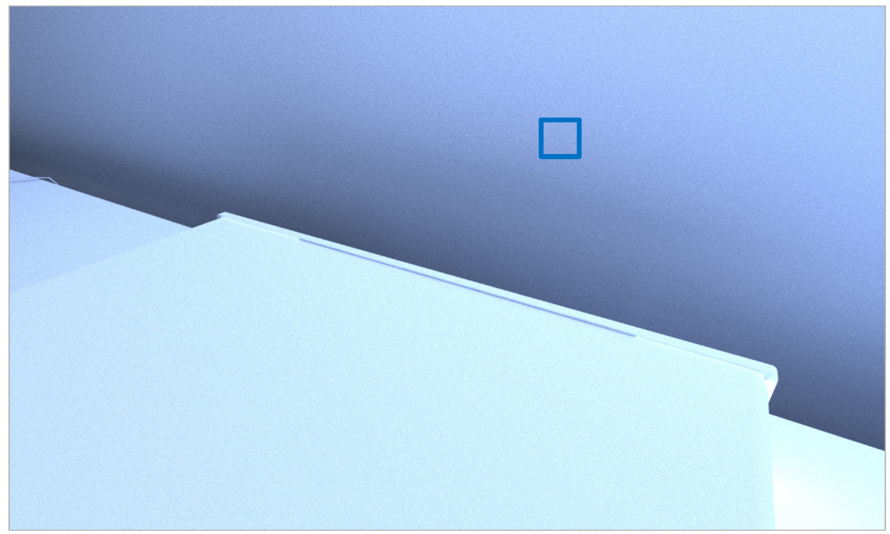
specular color, transparency, diffuse transmission, etc. deduced in a similar way

GI NEXT

Using the point clouds



piece-wise constant approximation



indirect lighting computed with GI Next

GI NEXT

Comparison to final gathering on a CPU

brute force final gathering

2 h 29 min

GI Next

1 h 13 min

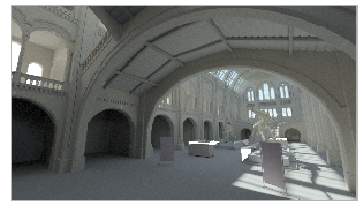
1280 x 720, 32 threads (16 cores)

GI NEXT

Taking advantage of the GPU

indirect illumination typically dominates computation time

1 min << 149 min



direct illumination only

direct + indirect illumination

GPU mode implemented with NVIDIA® OptiX™ as a plugin to mental ray®

- only scene geometry and point clouds need to be transferred to the GPU
- works with existing shaders, no need to port them (thanks to shader exploration)
- single BSDF model minimizes execution divergence on the GPU

GI NEXT ON THE GPU

Integration into the rendering process

- accelerate computation of batches of calls to core functionality
  - rendering
    - record calls to **`mi_compute_avg_radiance()`**
    - return a dummy value and discard shader results
  - batch computation of indirect illumination on the GPU
  - restart rendering with exact same initial state
    - shaders generate identical calls, this time receiving the results computed on the GPU
- requires repeatable rendering process: deterministic quasi-Monte Carlo

RESULTS

10x speedup or more on the GPU

brute force final gathering in 149 min (2h 29min)

32 CPU threads (16 cores)

GI Next GPU in 11 min 30s (~13x)

NVIDIA® Quadro™ GP100

RESULTS

10x speedup or more on the GPU

GI Next GPU in 2 min 25 sec (~48x)

NVIDIA® Quadro™ GP100

brute force final gathering in 117 min (1h 57min)

32 CPU threads (16 cores)
(scene courtesy of Lee Anderson)

RESULTS

10x speedup or more on the GPU

(video)

RESULTS

10x speedup or more on the GPU

brute force final gathering in 645 min (10h 45min)

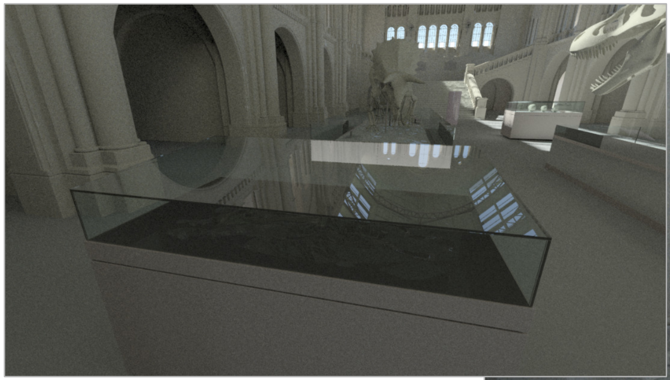
32 CPU threads (16 cores)

GI Next GPU in 1h (~10.7x)

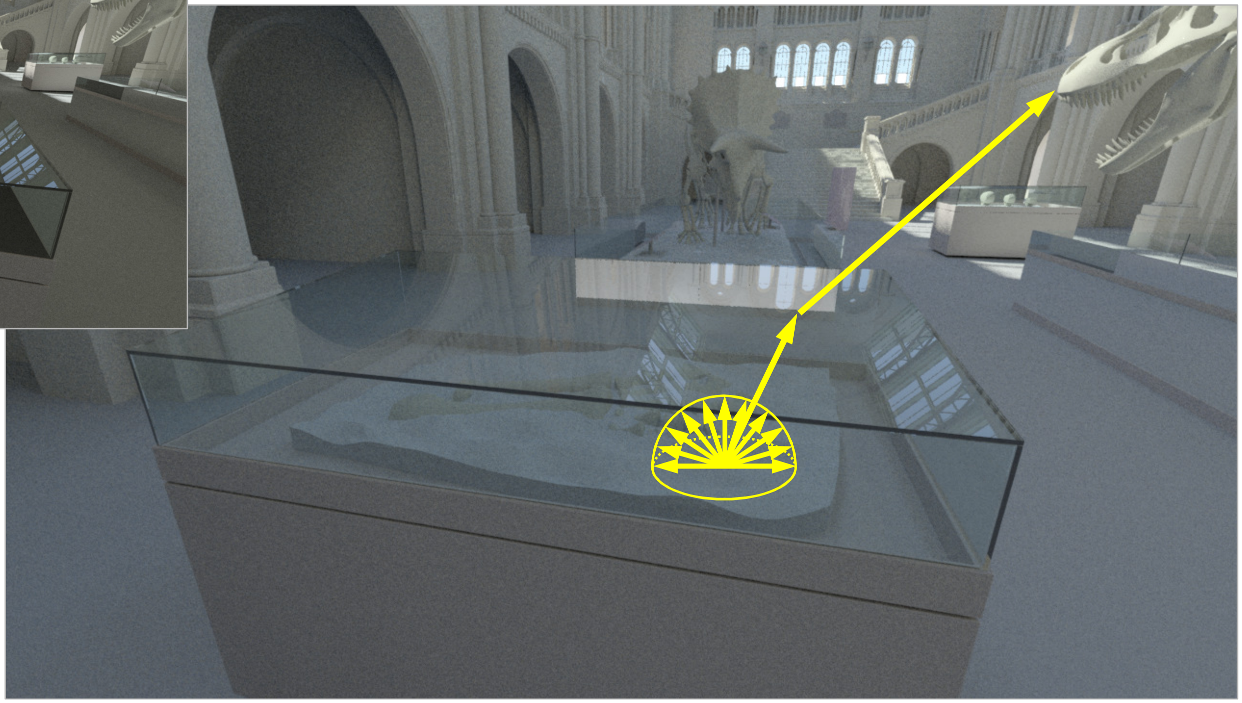
NVIDIA® Quadro™ GP100

RESULTS

Indirect illumination by path tracing using piecewise constant parameters



diffuse indirect illumination only



GI NEXT

Global illumination for production rendering on the GPU

path tracing on original geometry using one single generic BSDF

- exact visibility, full support for effects like motion blur and depth of field
- no shader calls, no access to texture data and no light loops
- simple controls: density of the point cloud, number of rays and depth
  - all of which may be controlled with a single quality value

more than 10x faster on a GPU using any legacy shaders

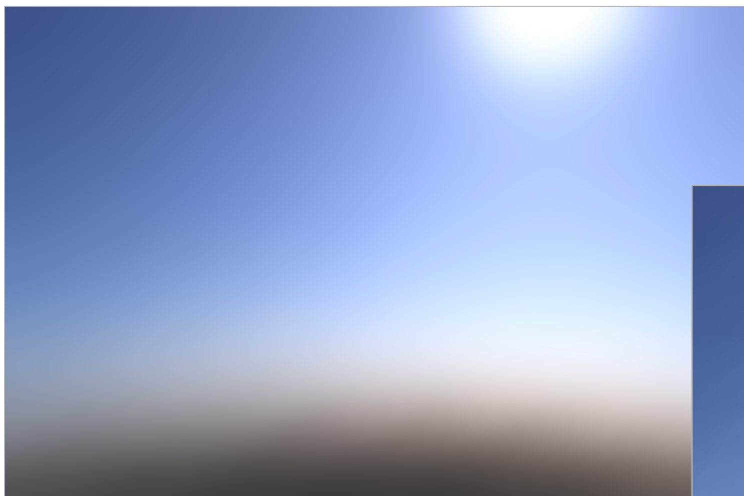
{ecatalano/rschoffel/kdahm/nbinder/akeller} @nvidia.com

MORE RESULTS

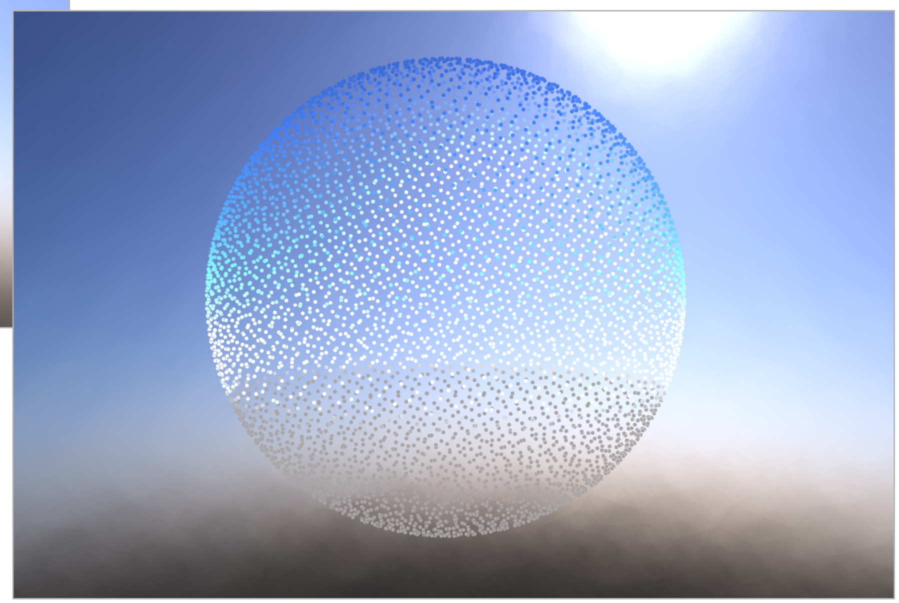
direct light only

GI NEXT

Creating point clouds



environment



environment shader characteristics stored in a second point cloud on a sphere