Enabling Real-Time Light Baking Workflows in Saber Engine with AMD Radeon™ Rays Library

Dmitry Kozlov, Advanced Micro Devices, Inc.

Max Gridnev, Saber Interactive

AGENDA

- Lightmapping basics
- Previous solution
- Radeon™ Rays
- Distributed baking
- Comparison of CPU/GPU solutions

Saber engine

WORLD WAR Z

QUAKE CHAMPIONS

HALO: THE MASTER CHIEF COLLECTION

R.I.P.D.

INVERSION

BATTLE LA

World War Z

Quake Champions

Halo: The master chief

R.I.P.D.

Lightmapping basics

Lightmap format

- Radiosity Normal Map, GI + Baked Lights
- Direction to the dominant light
- Radiance from the dominant light
- Reflection correction coefficient

Lightmap format

- Radiosity Normal Map, GI + Baked Lights
- Direction to the dominant light
- Radiance from the dominant light
- Reflection correction coefficient

Lightmap format

- Radiosity Normal Map, GI + Baked Lights
- Direction to the dominant light
- Radiance from the dominant light
- Reflection correction coefficient

Lightmap format

- Radiosity Normal Map, GI + Baked Lights
- Direction to the dominant light
- Radiance from the dominant light
- Reflection correction coefficient

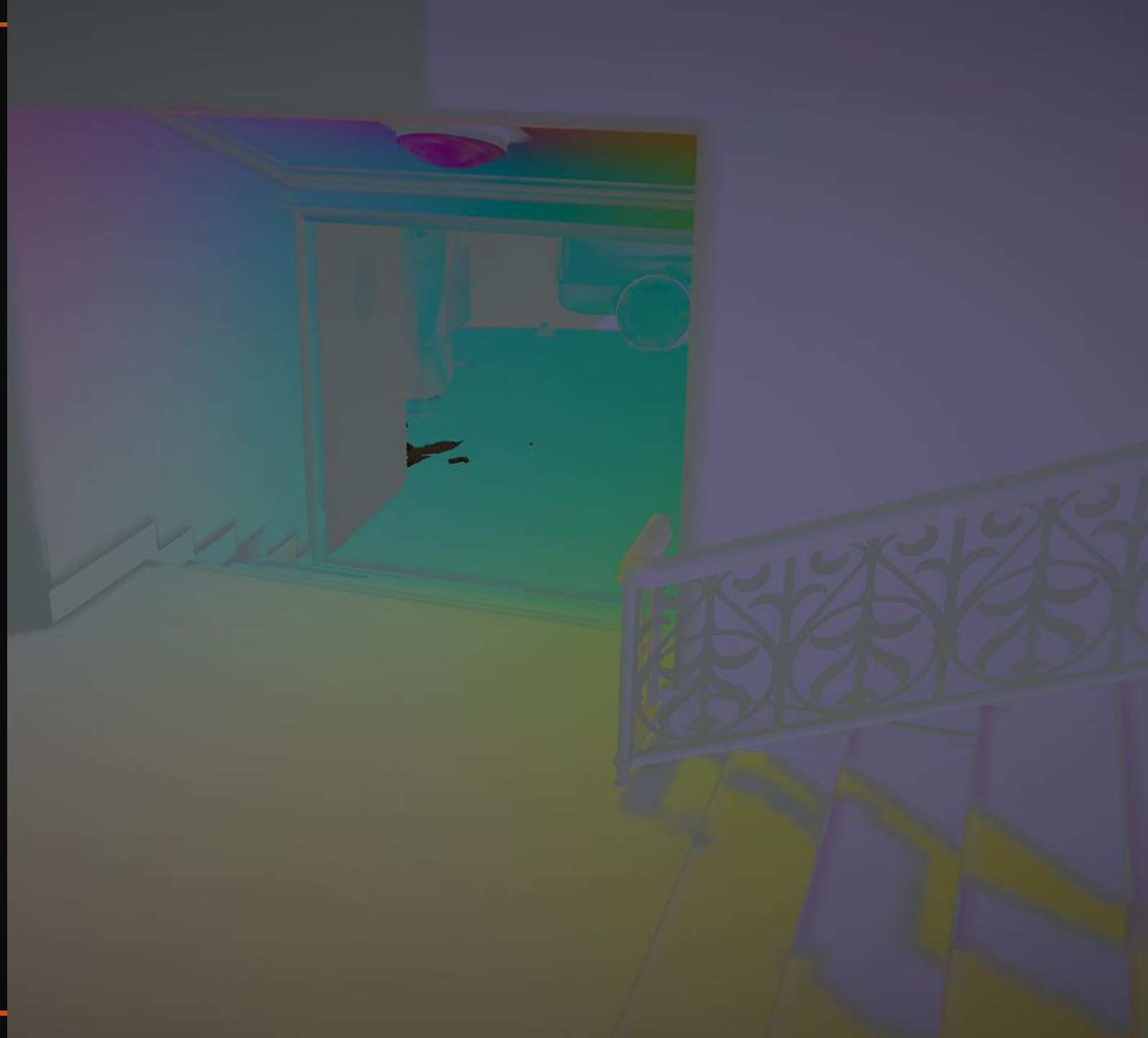
Directional light map

- 1 light source per lightmap texel
- Nice and cheap specular highlights
- Better than radiosity normal map (RNM) for extreme angles
- Great for low quality

Diffuse lighting

Dominant light radiance

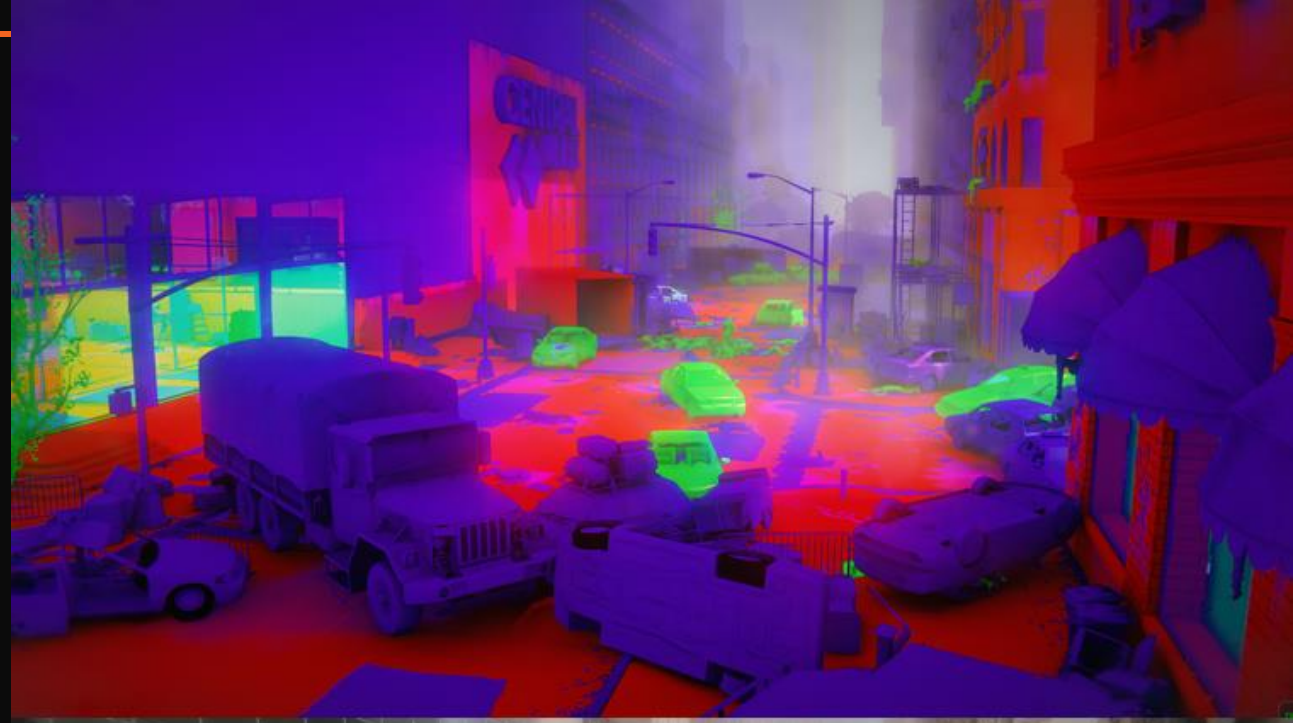
Dominant light direction

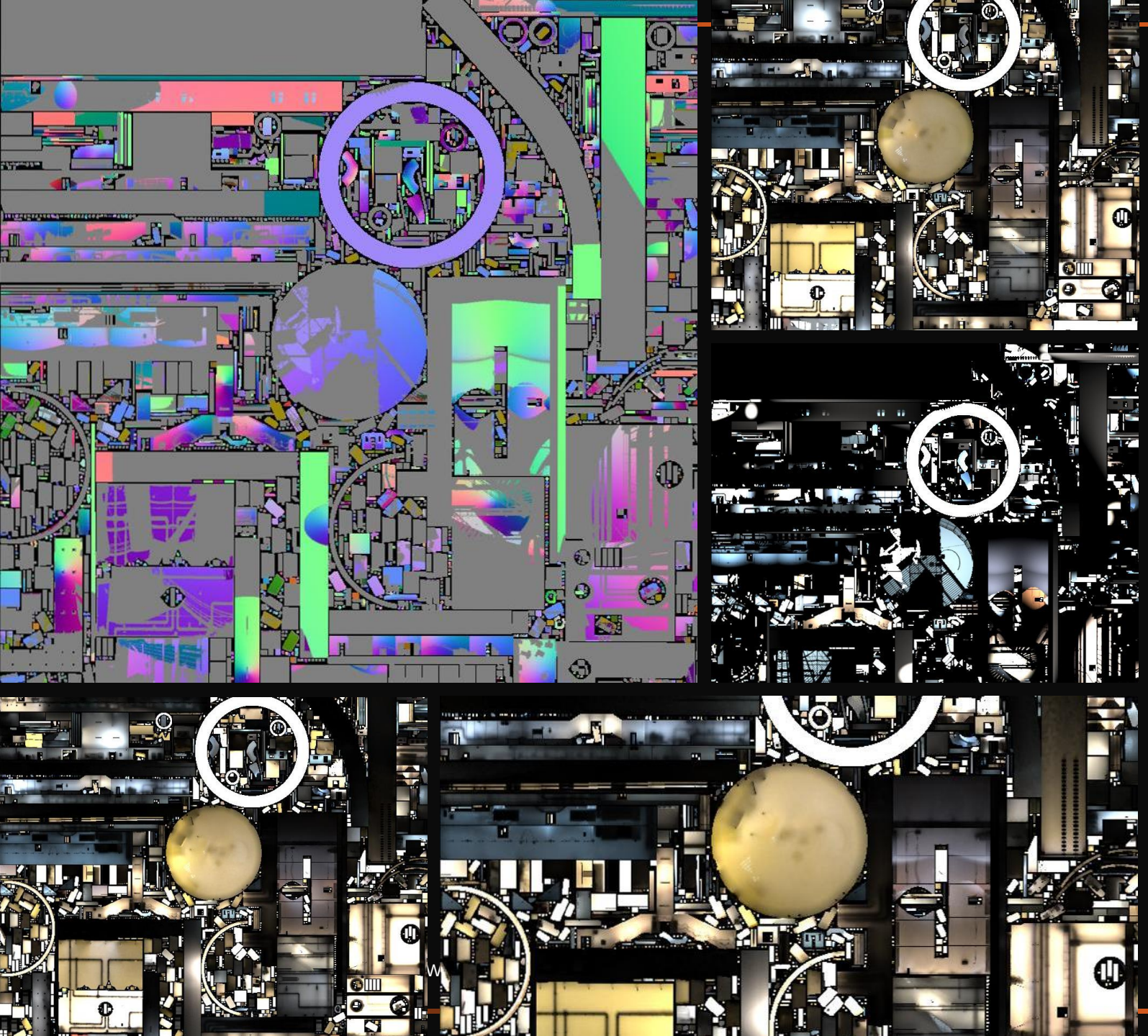


Full shading

Types of baking

- Texture lightmap
- Vertex lightmap
- Point cloud





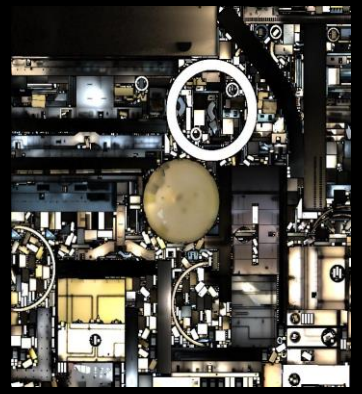
Packing lightmap data

Packed into 4 BC6 textures

- RNM (3 textures)
- Dominant radiance (1 texture)

Single BC7 texture

- RGB: dominant light direction
- Alpha: reflection coefficient



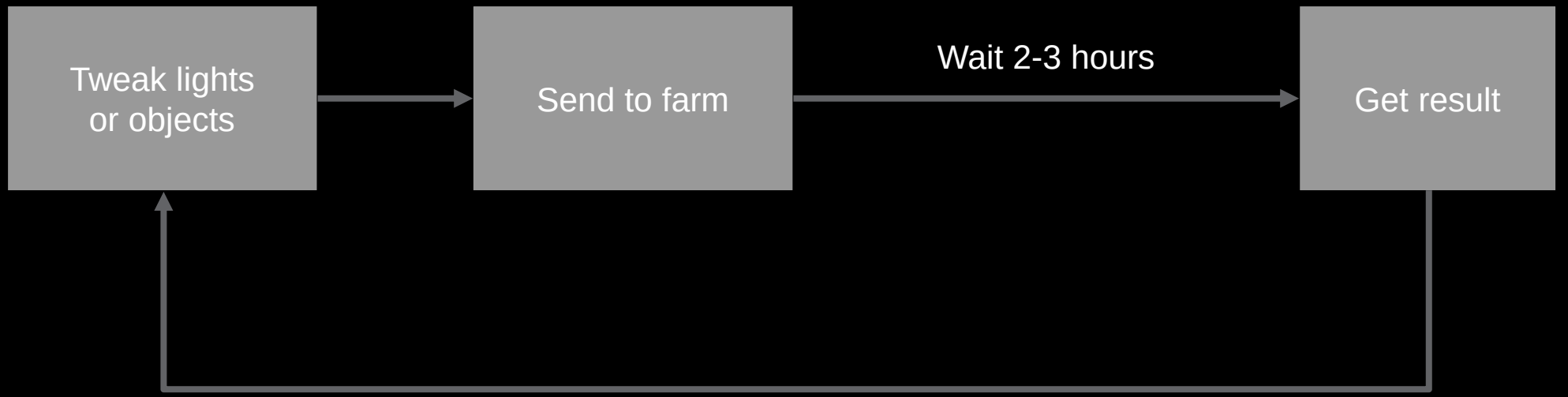
Packing lightmap data

- Vertex points clustered by similarity and packed into a group of 4x4
- Works with BC compression

Point cloud

- Volumes placed by artist
- Consist of tetrahedrons
- Bake lighting at points
- Used for dynamic objects
- Data
  - SH2 for diffuse
  - Direction + Radiance for specular highlights

Lighting artist workflow



Previous light baking solution in Saber engine

Based on Autodesk Beast

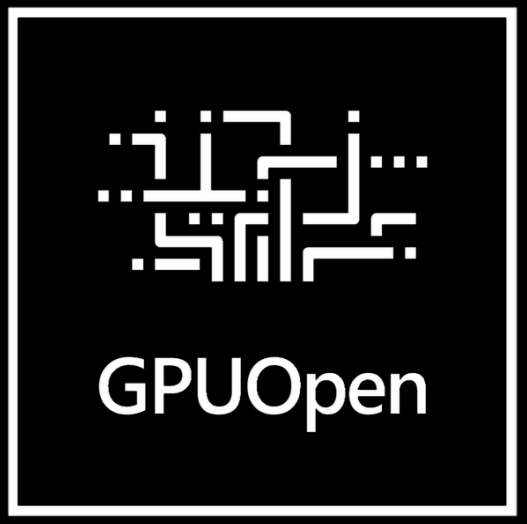
- CPU based => slow ~2h per level (1000 CPU cluster)
- Closed software
  - Engine specific features like attenuation function, projection texture, etc.
  - No real-time update for our format
- Support is non flexible and hard work

Path tracing basics

- Start from a lightmap texel
- Trace rays unidirectionally
- Accumulate direct and indirect light

Why Radeon™ Rays?

- Open source
- Hardware independent (OpenCL™)
- Has a path-tracing engine (Baikal)



Tweaking Radeon™ Rays

Inject into BVH traversal code

- Alpha kill texture masking
- LOD masking
  - For a texel’s object: skip all lods except starting lod
  - Skip all non high-level LOD for other objects

Radeon™ Rays 3.0 Design Features

- Vulkan 1.0 compatible
- Hardware independent
- Platform independent

Radeon™ Rays 3.0 Design Features

- Features and improvements
  - Low-level C API
  - GPU acceleration structure builds (both scenes and meshes)
  - Fast acceleration structure updates
  - New acceleration structure types
    - Regular and irregular grids
    - Hierarchical grids
    - Compressed BVHs
  - Rapid Packed Math support on Radeon™ Vega (FP16)
  - Many performance optimizations

Radeon™ Rays 3.x roadmap

- Support for new geometric primitives
  - Hair strands
  - NURBS surface patches
- Out of core geometry
- Optimized (on chip) traversal for alpha-tested geometry
- More complex BVH compression schemes

Noise reduction

- Bilateral filter (lightmap space/spatial)
  - Works fine because diffuse GI is low frequency
  - Average nearest texels within specified radius, taking normals, positions, and radiance into account
- ML filter (future direction)

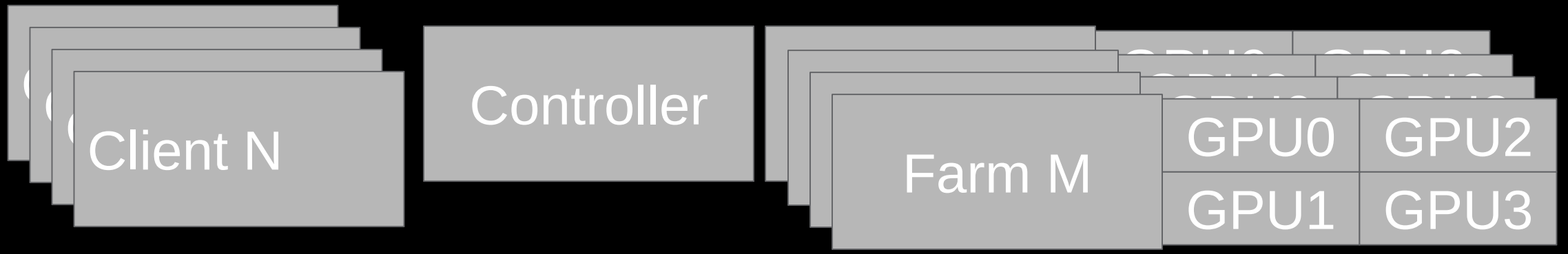
Radeon™ Image Filters Design Features

- Cross platform image processing library
- Hardware independent
- Conventional post processing filters:
  - Antialiasing
  - Tone mapping
  - Color space conversions
- Denoisers:
  - EAW, LWR, SVGF
  - ML denoiser (OpenCL™ & DirectML)
  - ML upscaler (OpenCL™ & DirectML)

Real-time preview

- Update only visible parts
- Trace rays from camera
- Filter all visible texels

Distributed baking



Distributed preview challenges

- **8 GB** of typical GPU memory limit
- Data distribution (measure on a typical level in Saber engine)
  - **2.5 GB** BVH
  - **2 GB** auxiliary data
  - ~**3 GB** lightmap data (4K)

Solution: update only visible texels and readback to system memory

(*) Information provided by Saber Interactive

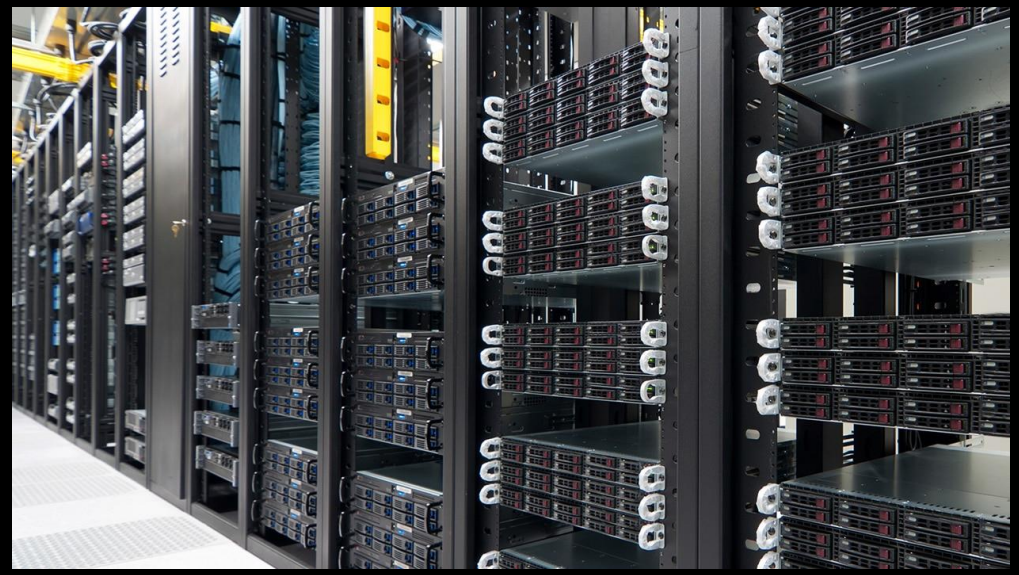
Comparison

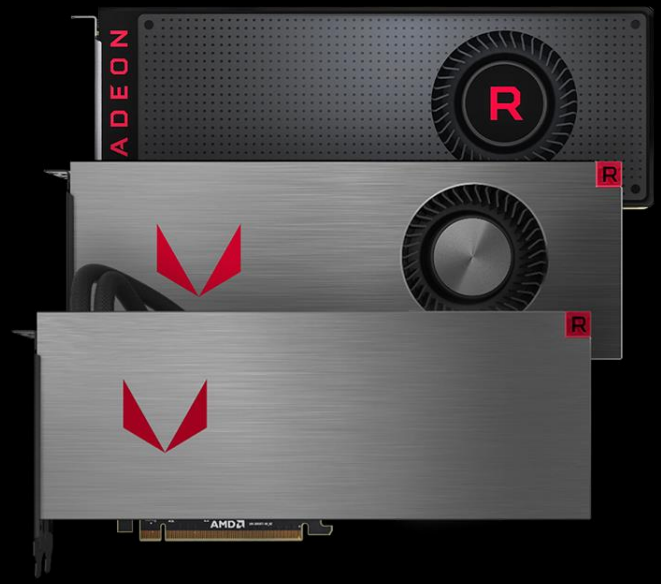
- Typical level
  - 15m triangles
  - 4Kx2K texture lightmap
  - 3Kx3K vertex lightmap
  - 200K point cloud

Comparison

CPU cluster: 64x Intel CPU Xeon E7-8870 (10 cores x 2 threads, 64GB RAM per CPU), Windows 10 x64

GPU cluster: 1x Intel CPU Xeon E7-8870 (64 GB RAM) + 2x Radeon™ Vega 64 GPU (8GB VRAM), Windows 10 x64

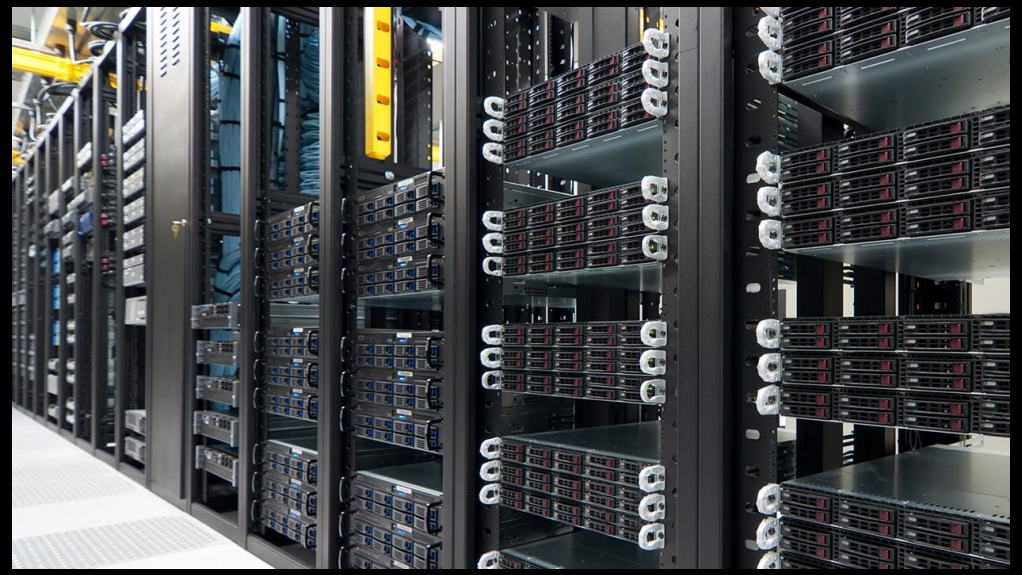


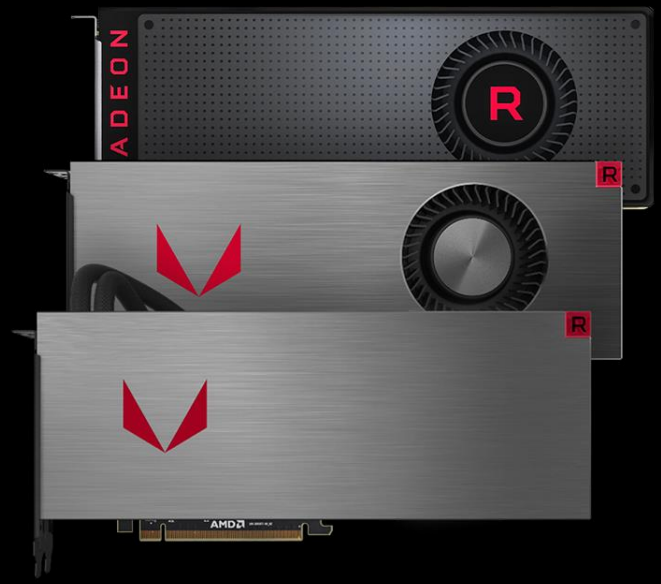


Comparison

CPU cluster: ~2 h

GPU cluster: ~20 min





(*) Testing done by Maxim Gridnev January 15, 2019. PC manufacturers may vary configurations yielding different results. Results may vary based on driver versions used.

Distributed baking

Pros

- Much faster end-to-end baking time compared to CPU cluster
- Much more better perf / $ compared to CPU cluster
- Designed to enable new baking workflows (place & edit)

Cons

- Latency ~0.5 sec (see future directions)
- CPU only filter (can be potentially ported to GPU)

Conclusion

Implemented distributed lightmap baking service using Radeon™ Rays

Benefits

- Cheaper and faster
- Enables new workflows

Future work

- Balancing schemes for render farms
- Faster mGPU and CPU-GPU transfer schemes
- ML noise filtering
- Data compression schemes
- Geometry updates

Disclaimer and attributions