SIGGRAPH 2021

Radiance Caching for real-time Global Illumination

Daniel Wright - Epic Games
@EpicShaders

Final Gather techniques used in Lumen

Lumen: Dynamic Global Illumination in Unreal Engine 5

Targeting games on next-generation consoles


Scaling up to quality-first Enterprise
on high end PC

Background

Global Illumination

$$Lo(v) = Le(v) + \int Li(l) fs(l \to v) cos(\Theta l) dl$$

What we are solving for every pixel

Global Illumination

$$\boxed{Lo(v)} = Le(v) + \boxed{\int} \boxed{Li(l)} \boxed{fs(l \rightarrow v)} cos(\Theta l) dl$$

Final pixel lighting is the Integral over incoming radiance times the BRDF

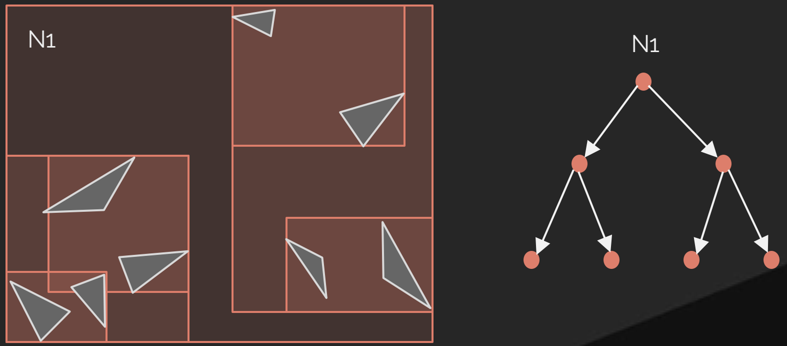
Monte Carlo Integration

$$\lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \frac{\boxed{Li(l)} fs(l \to v) cos(\Theta l)}{Pk}$$

- Estimate with discrete samples
- Find incoming radiance in a direction with Ray Tracing

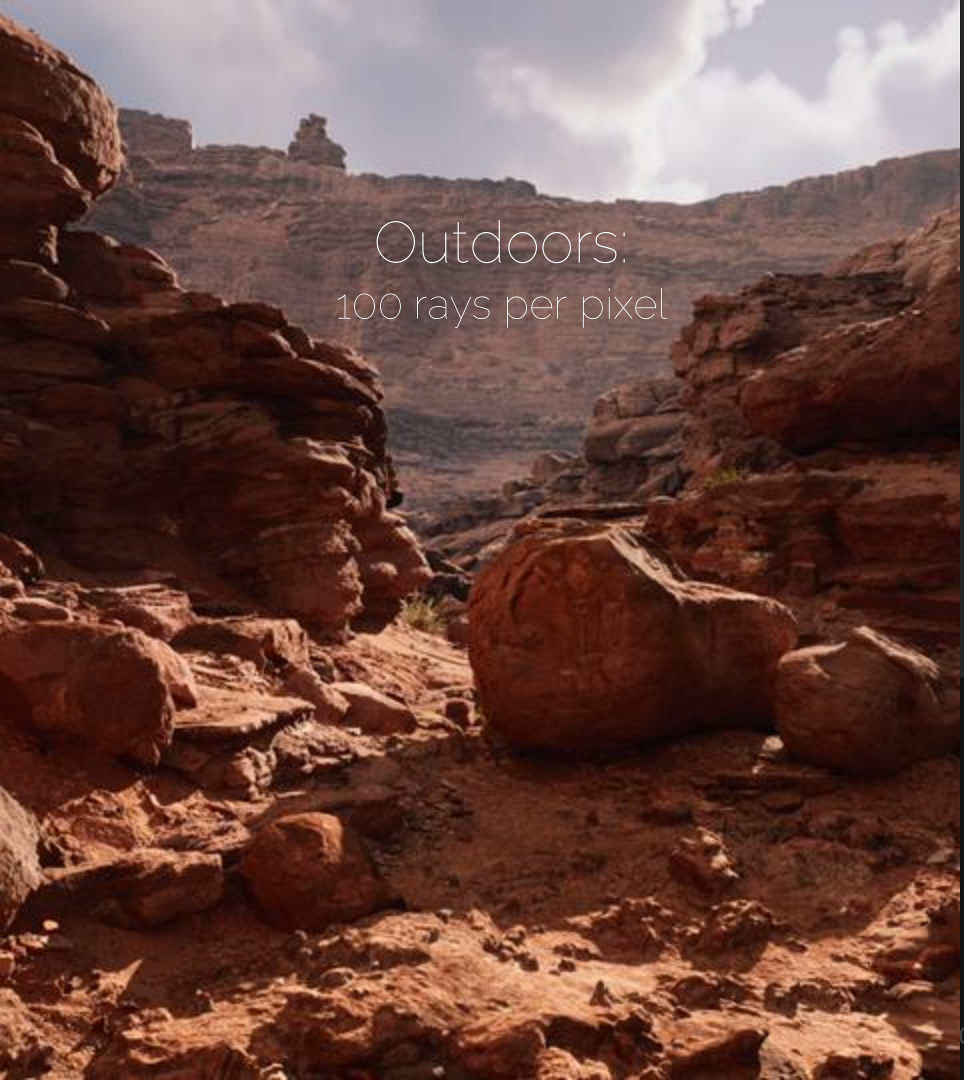
Ray Traces are slow

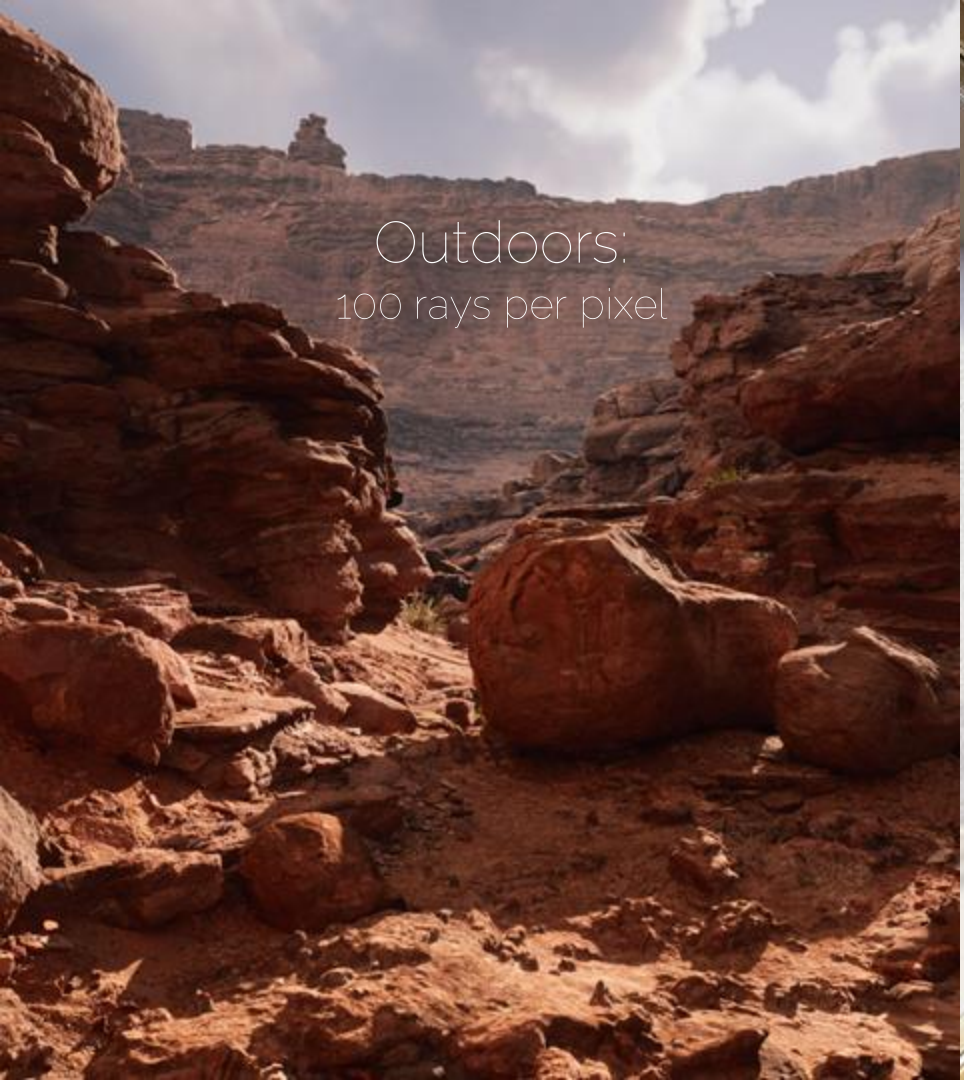
- Two level BVH
  - Incoherent tree traversal
  - Instance overlap



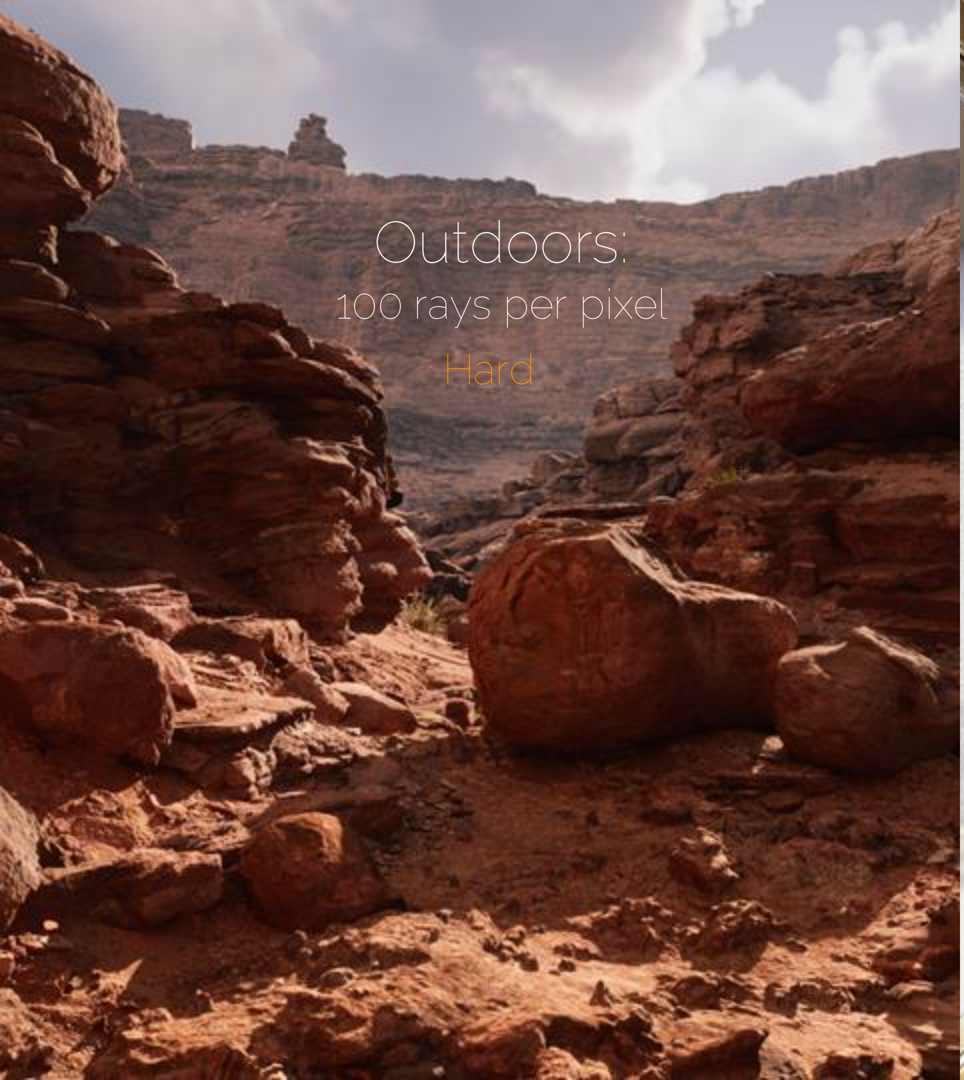
Ray Traces are slow

- Can only afford 1/2 ray per pixel
- But quality GI needs hundreds!


Outdoors:
100 rays per pixel


Outdoors:
100 rays per pixel


Indoors:
500+ rays per pixel


Outdoors:
100 rays per pixel
Hard


Indoors:
500+ rays per pixel
Extremely hard

How can this be made realtime?

Previous real-time work: Irradiance Fields

- Ray Trace from a small set of probes
  - Arranged in world space grids [Tatarchuk 2012]
- Pre-calculate Irradiance
- Interpolate to full resolution pixels
- Probe Occlusion to reduce leaking [Valient 2014] [McGuire 2019]

Irradiance Field problems

- Leaking and over-occlusion
- Probe placement
- Slow lighting update
- Distinctive flat look
  - Irradiance near occluders is higher frequency than spatial probe resolution

Previous real-time work: Screen Space Denoiser

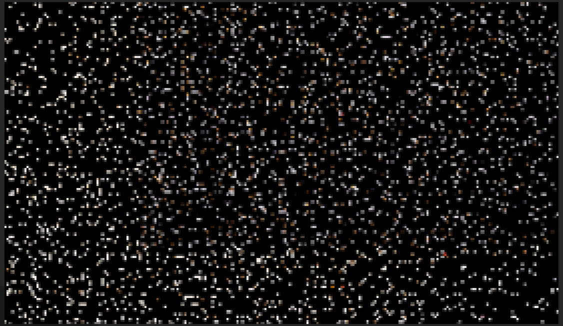
- Ray Trace from pixels
  - Cos distribution
  - ~1 ray per pixel
- Denoise with spatial and temporal reuse
  - Spatiotemporal Variance-Guided Filtering [Schied et al 2017]
  - Fast Denoising with Self Stabilizing Recurrent Blurs [Zhdan 2020]

Screen Space Denoiser problems

- Input is too noisy - fixed sample rate

Noise increases as bright feature becomes smaller

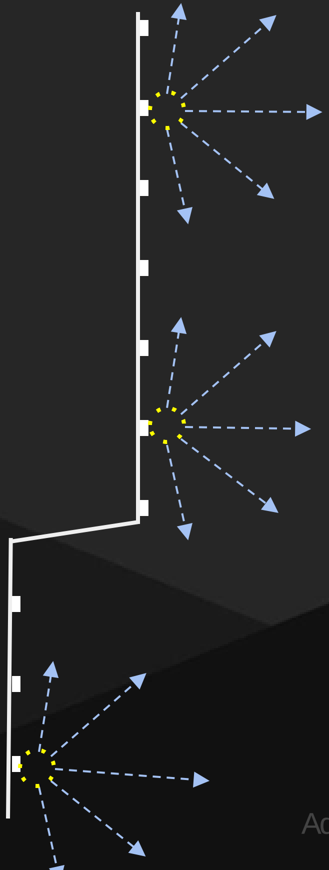
Near bright window



Far from window

Our approach

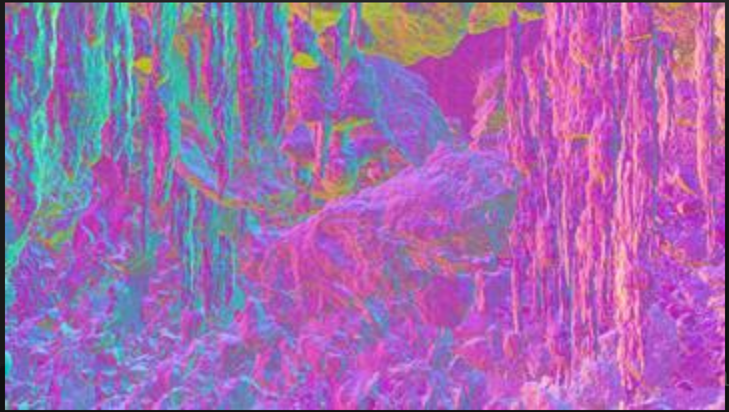
Screen Space Radiance Caching



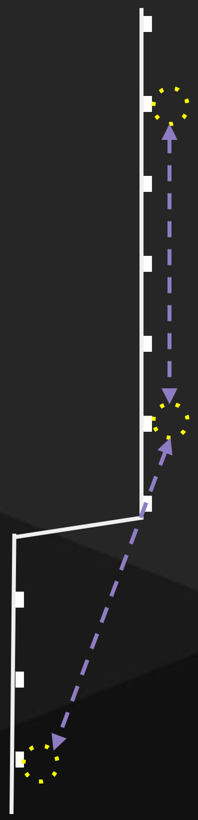
Downsample incoming radiance

- Incoming light is coherent, geometry normals are not

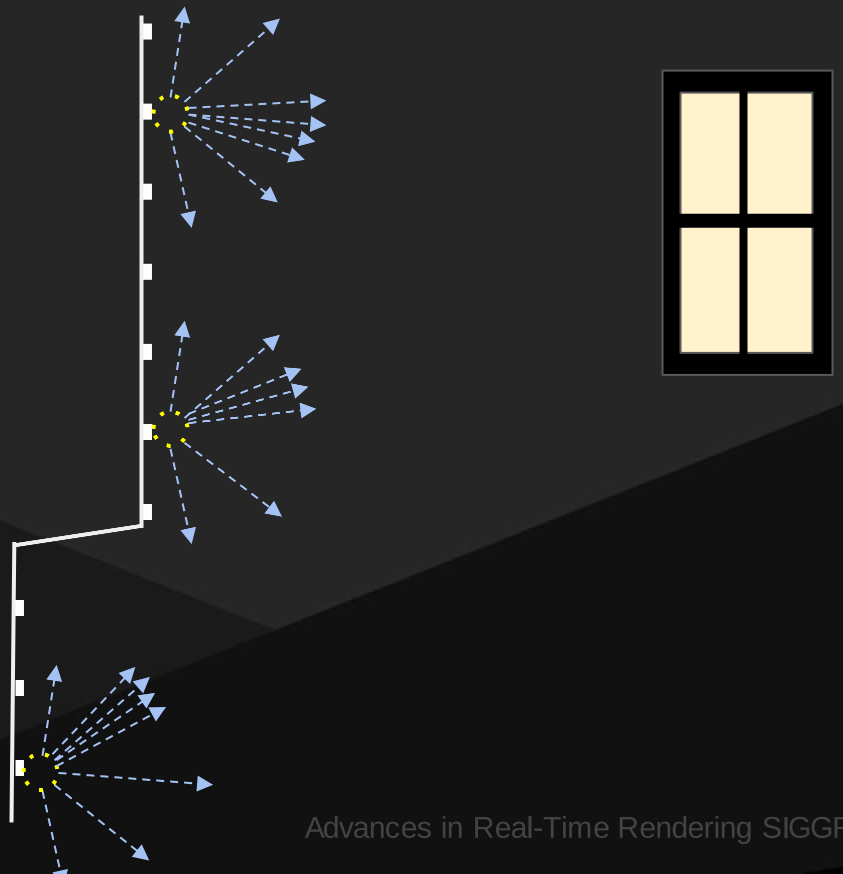
Integrate incoming lighting over the BRDF at full res



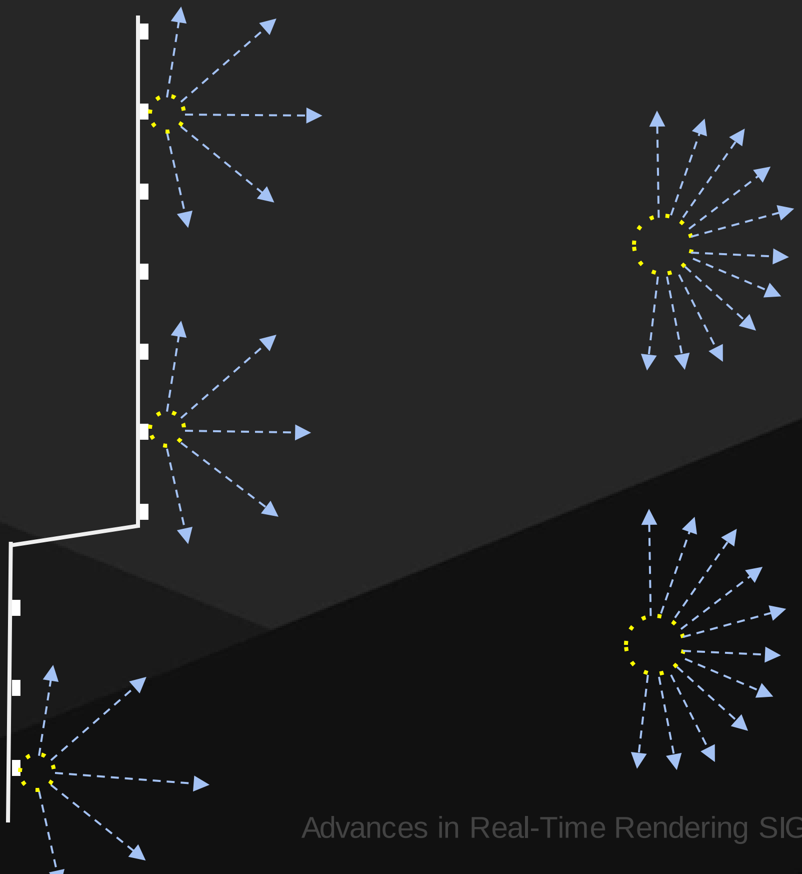
Filter in radiance cache space, not screen space



Better sampling in the first place - importance sample incoming lighting

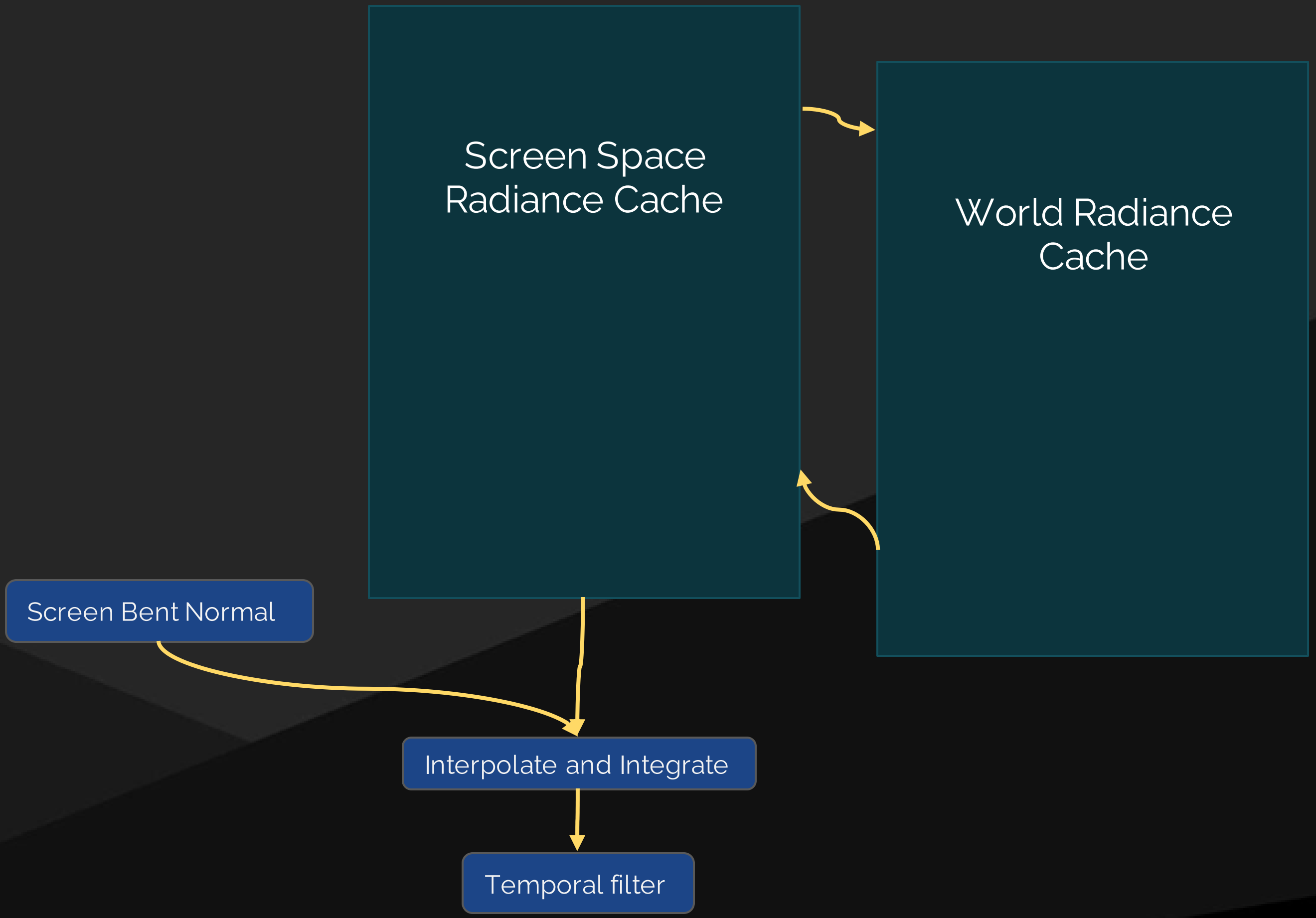


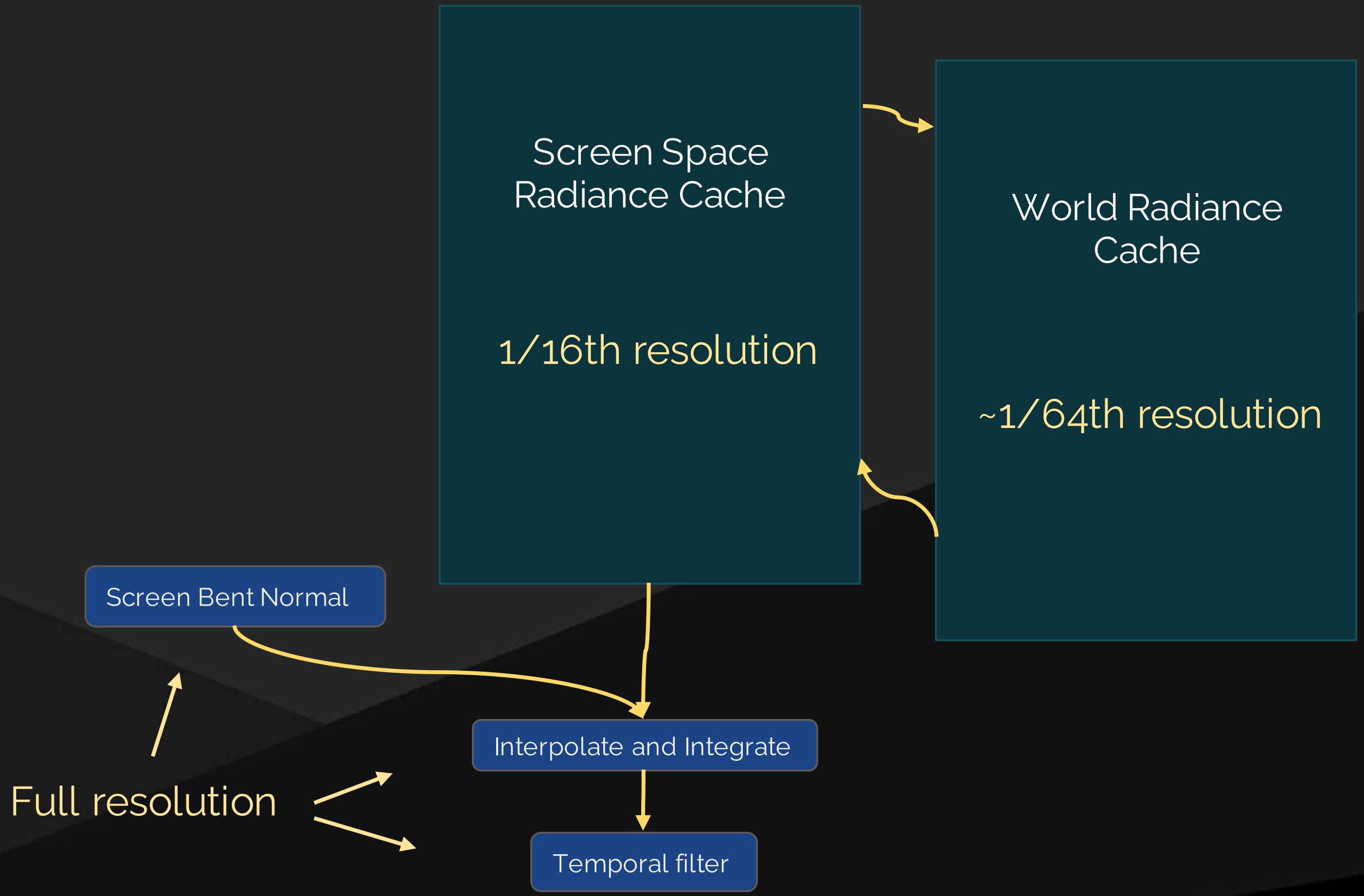
Stable distant lighting with World Space Radiance Caching

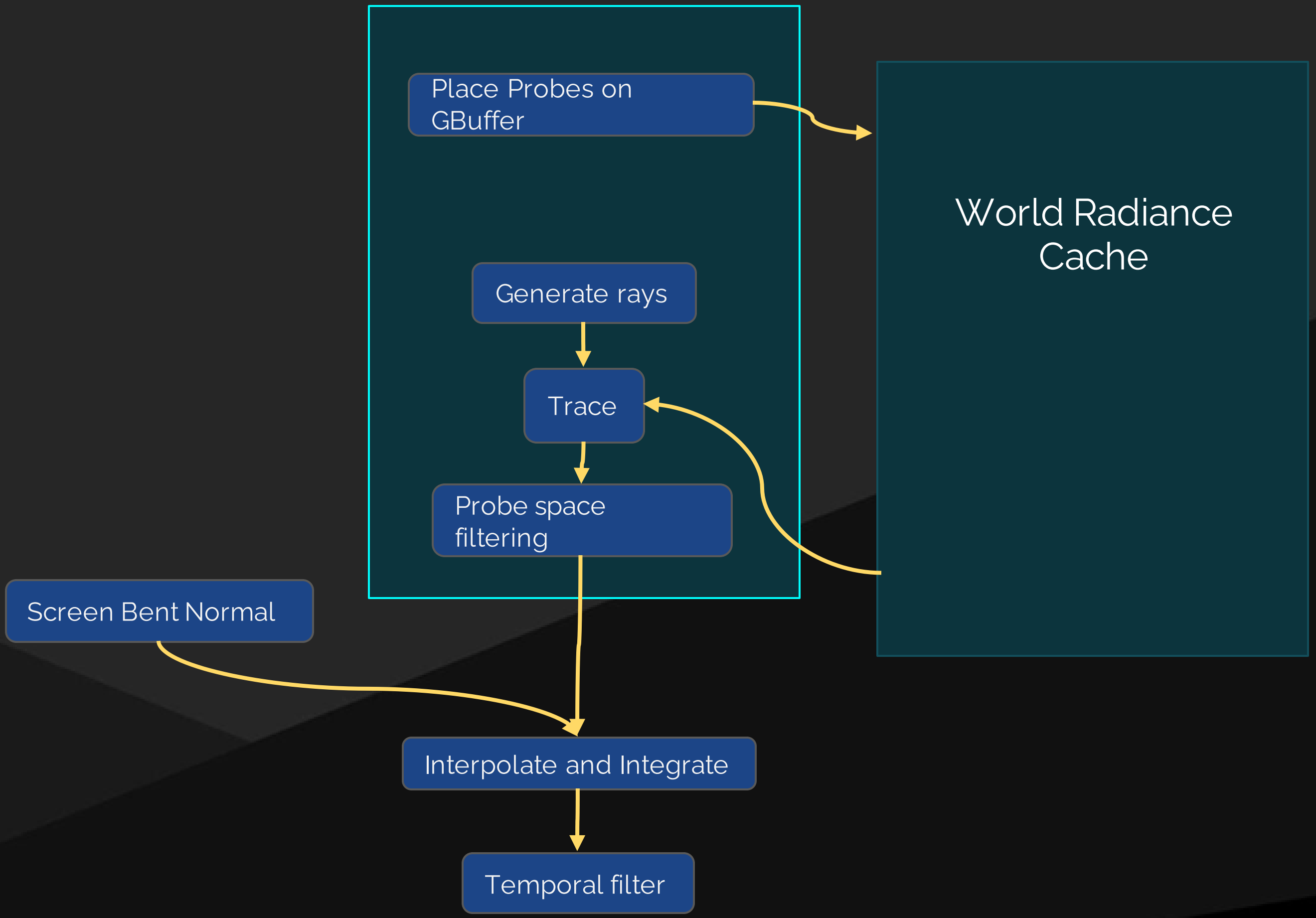


Raw input compare

Final Gather Pipeline

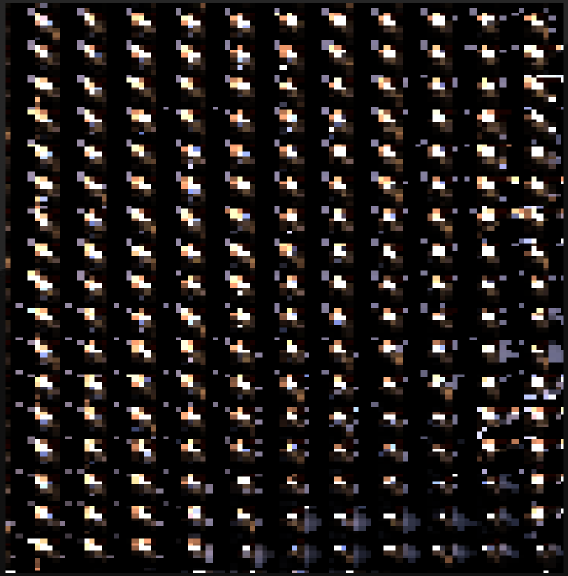

Screen Space Radiance Cache
World Radiance Cache
Screen Bent Normal
Interpolate and Integrate
Temporal filter


Screen Space Radiance Cache
1/16th resolution
World Radiance Cache
~1/64th resolution
Screen Bent Normal
Interpolate and Integrate
Temporal filter
Full resolution


Place Probes on GBuffer
Generate rays
Trace
Probe space filtering
World Radiance Cache
Screen Bent Normal
Interpolate and Integrate
Temporal filter

Screen Probe structure

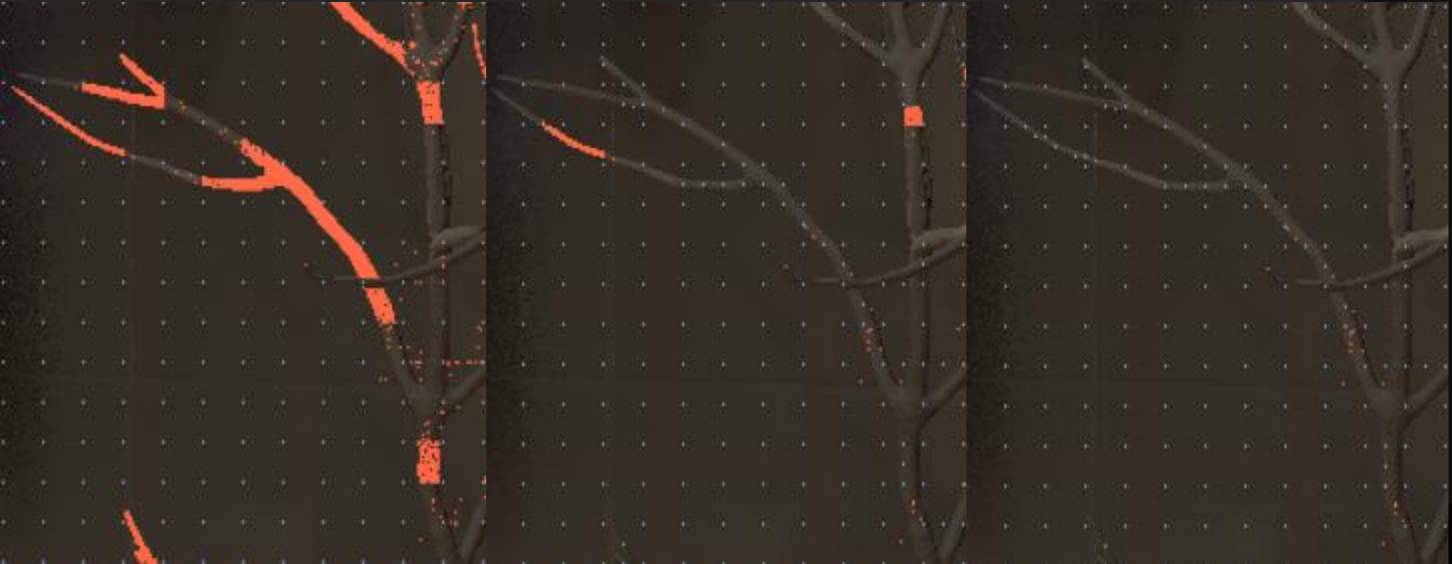
- Octahedral atlas with border
  - Typically 8x8 per probe
  - Uniformly distributed world space directions
  - Neighbors have matching directions
- Radiance and HitDistance in 2d atlas



Screen Probe placement

- Adaptive placement with Hierarchical Refinement [Křivánek et al 2007]
  - Iteratively place where interpolation fails

16 8 4



Screen Probe placement

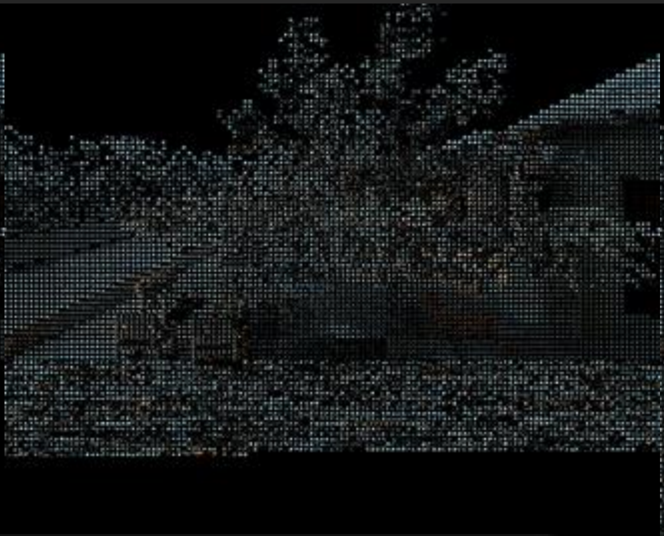
- Adaptive placement with Hierarchical Refinement [Křivánek et al 2007]
    - Iteratively place where interpolation fails
- Flood fill for final level

Adaptive sampling

- Need upper limit for real-time
- Don't want extra barriers for processing adaptive probes
  - Place adaptive probes at the bottom of the atlas

16

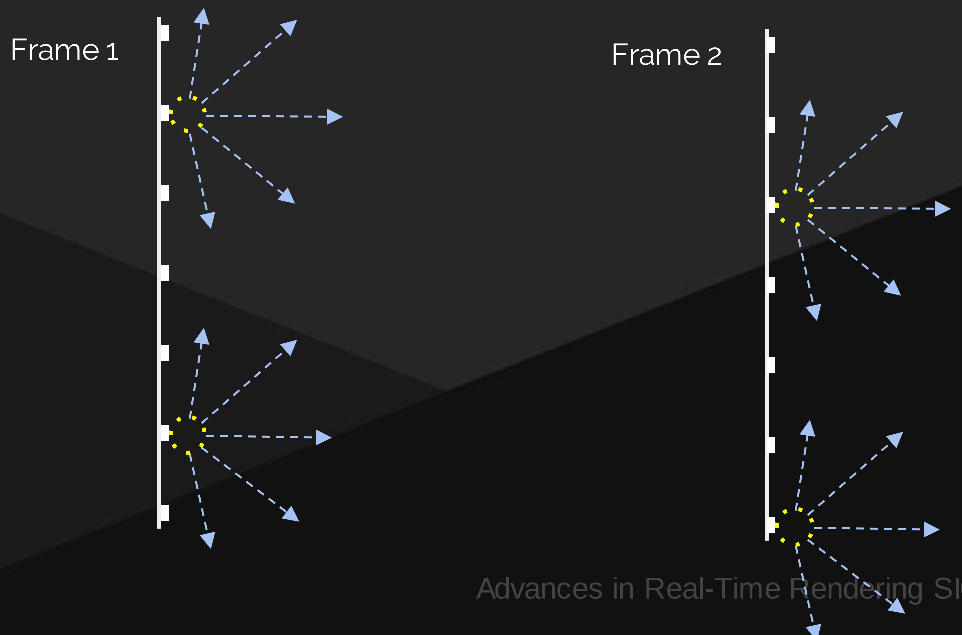
8



4

Screen Probe jittering

- Temporally jitter placement grid and direction
- Place directly on pixels
  - No leaking
  - Occlusion differences within screen cell have to be hidden with temporal filter



Interpolation

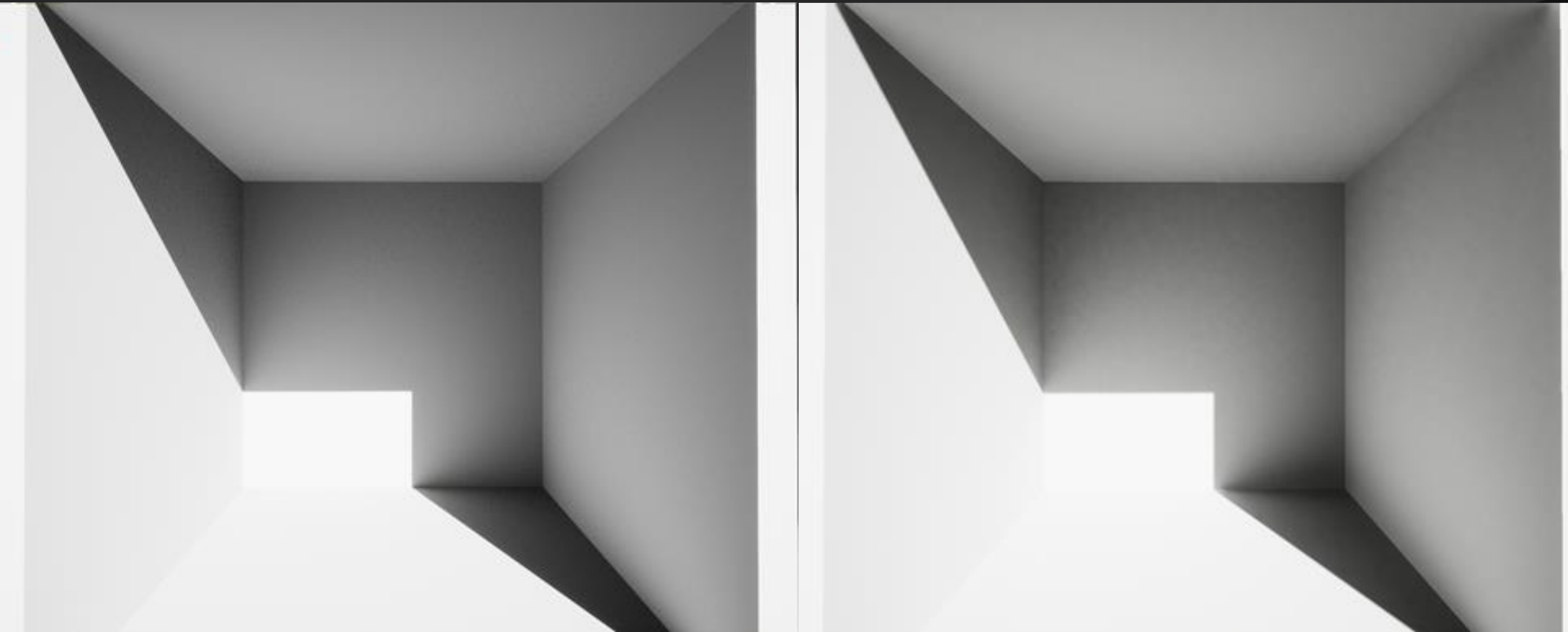
- Plane distance weighting
  - Prevents foreground misses leaking onto background

Interpolation

- Plane distance weighting
  - Prevents foreground misses leaking onto background
- Jitter offset into interpolation
  - *only if still in same plane
  - Distributes differences between probes spatially
  - Temporally stabilizes final lighting by expanding TAA 3x3 neighborhood

Screen Space Radiance Cache pipeline validation

- Matches path tracer when cranked up



But too much noise at ½ ray per pixel

Importance Sampling

$$\lim_{N \to \infty} \frac{1}{N} \sum_{k=1}^{N} \frac{\boxed{Li(l)}\boxed{fs(l \to v)cos(\Theta l)}}{Pk}$$

- We would like to distribute rays proportional to the integrand
- How can we estimate these?

$$\lim_{N\to\infty} \frac{1}{N}\sum_{k=1}^{N} \frac{\boxed{Li(l)}\, fs(l \to v) cos(\Theta l)}{Pk}$$

- Incoming Radiance:
    - Reproject last frame's Screen Probe Radiance!
        - No need to do an expensive search
        - Rays already indexed by position and direction
    - Fallback to World Space Probe Radiance on disocclusion

$$\lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \frac{Li(l)\boxed{fs(l \to v)cos(\Theta l)}}{Pk}$$

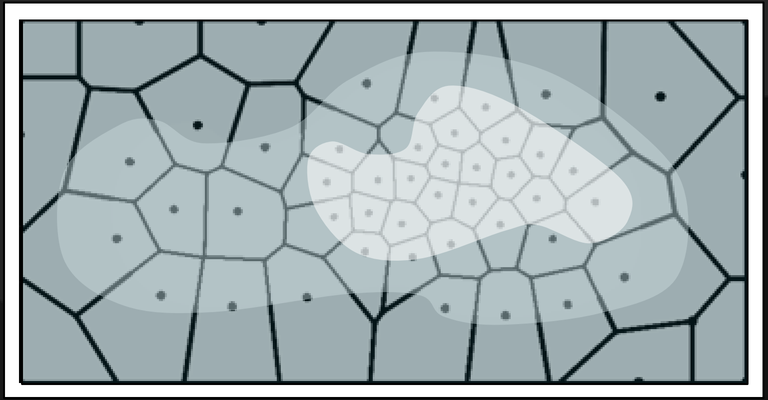
- BRDF:
  - Accumulate from pixels that will use this Screen Probe

$$\lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \frac{\boxed{Li(l)fs(l \to v)cos(\Theta l)}}{Pk}$$

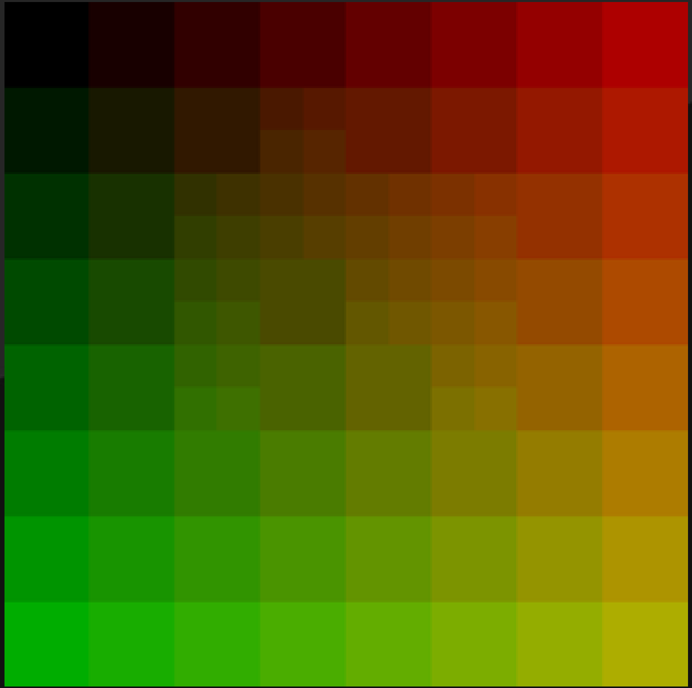
- Even better, we would like to sample proportional to the product of the incoming Radiance and BRDF

Structured Importance Sampling

- Assigns a small number of samples to hierarchically structured areas of the Probability Density Function (PDF) [Agarwal et al 2003]
  - Achieves good global stratification
  - Sample placement requires offline algorithm

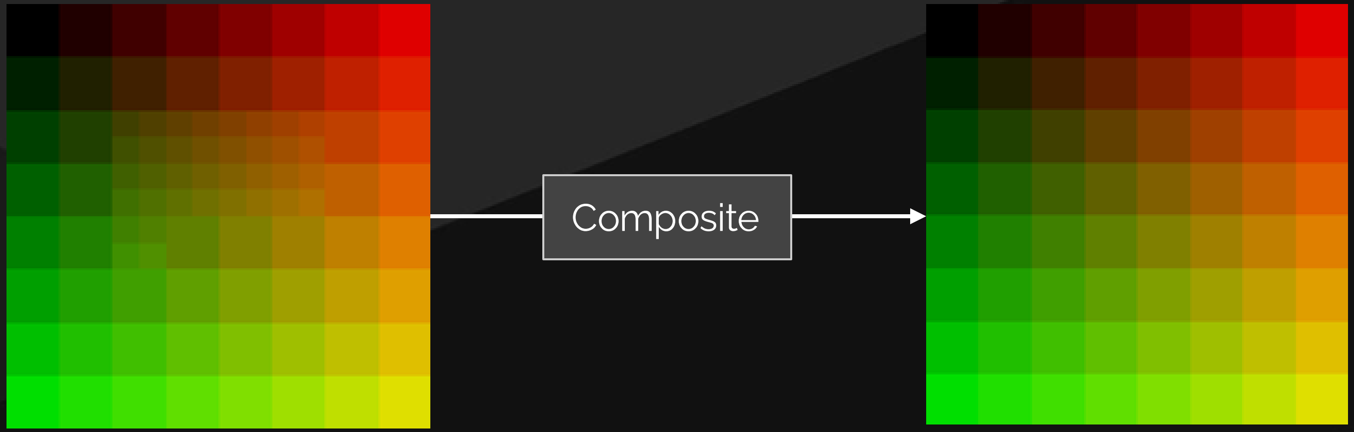


Maps perfectly to Octahedral mip quadtree!



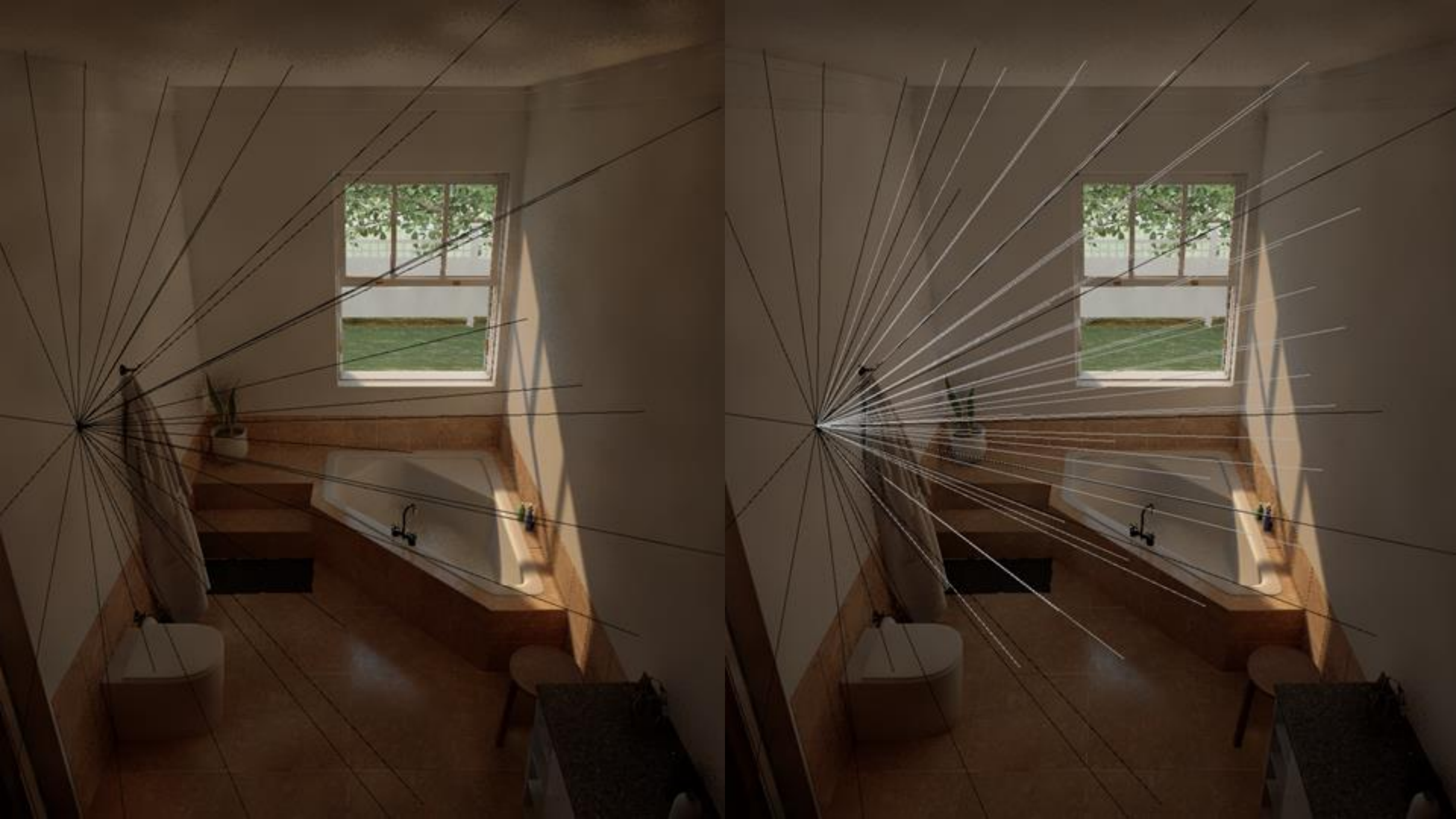
Integrating into pipeline

- Add indirection to tracing lanes
  - Store RayCoord, MipLevel
- After tracing, composite TraceRadiance into uniform probe layout for final integration



Ray Generation algorithm

- Calculate BRDF PDF * Lighting PDF for each Octahedral texel
- Start with uniformly distributed probe ray directions
  - Want fixed output ray count - keep tracing lanes full
- Sort rays by PDF
- For every 3 rays with PDF below cull threshold, supersample matching high PDF ray

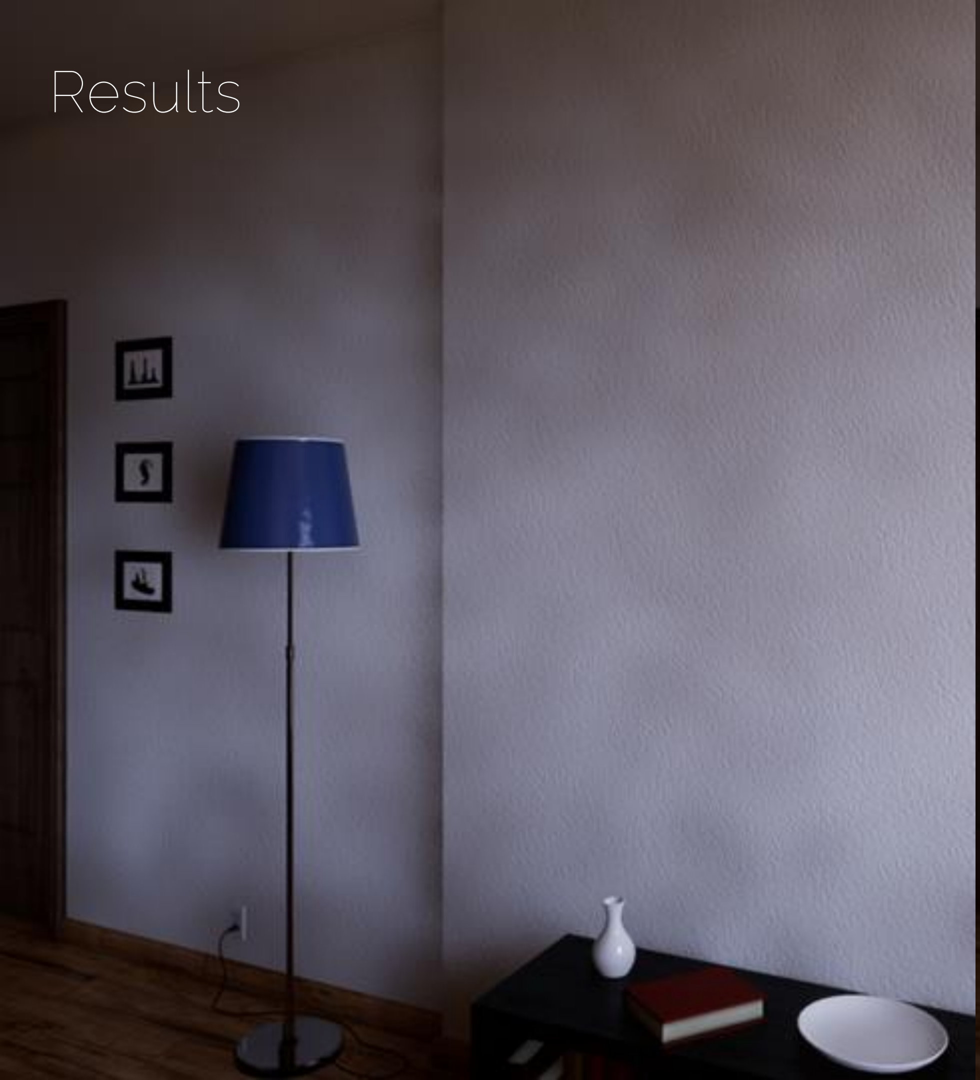


Improvements

- Don't allow Lighting PDF to cull rays
  - Lighting PDF is approximate, BRDF is accurate
- Can cull more aggressively by leaning on spatial filter
  - Cull with higher BRDF threshold
  - Reduce weight of culled rays during spatial filter
    - Fixes darkening around corners

Results

Results



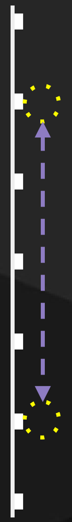
Importance Sampling recap

- Guide this frame's rays with last frame's lighting, and distant lighting
- Bundling rays into probes lets us afford smarter sampling

Spatial filtering

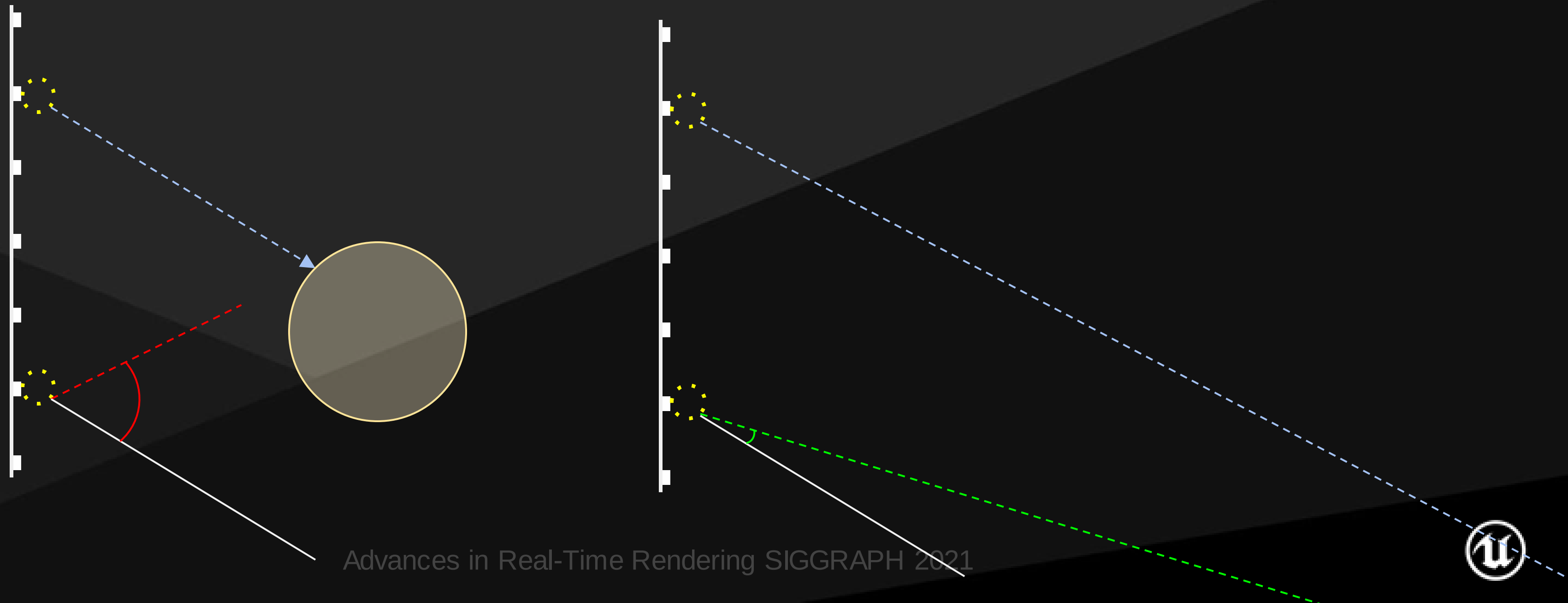
Filtering in Radiance Cache space

- Large spatial filter for cheap
  - 3^2 in probe space vs 48^2 screen space
- Can ignore normal differences between spatial neighbors
  - Only depth weighting



Gather Radiance from neighbors

- Gather from matching Octahedral cell in neighbor probes
- Error weighting:
  - Angle error from reprojected neighbor ray hits
  - Filters distant lighting, preserves local shadowing



Preserving contact shadows

- Angle error biases toward distant light = leaking
    - Distant light has no parallax and never gets rejected
- Solution: clamp neighbor hit distance to our own before reprojection

Final filtering compare

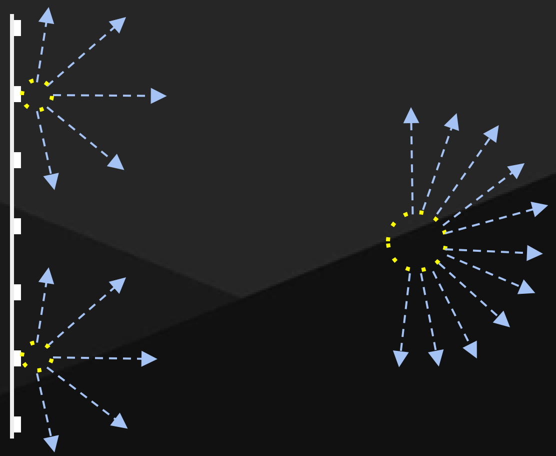
World Space Radiance Cache

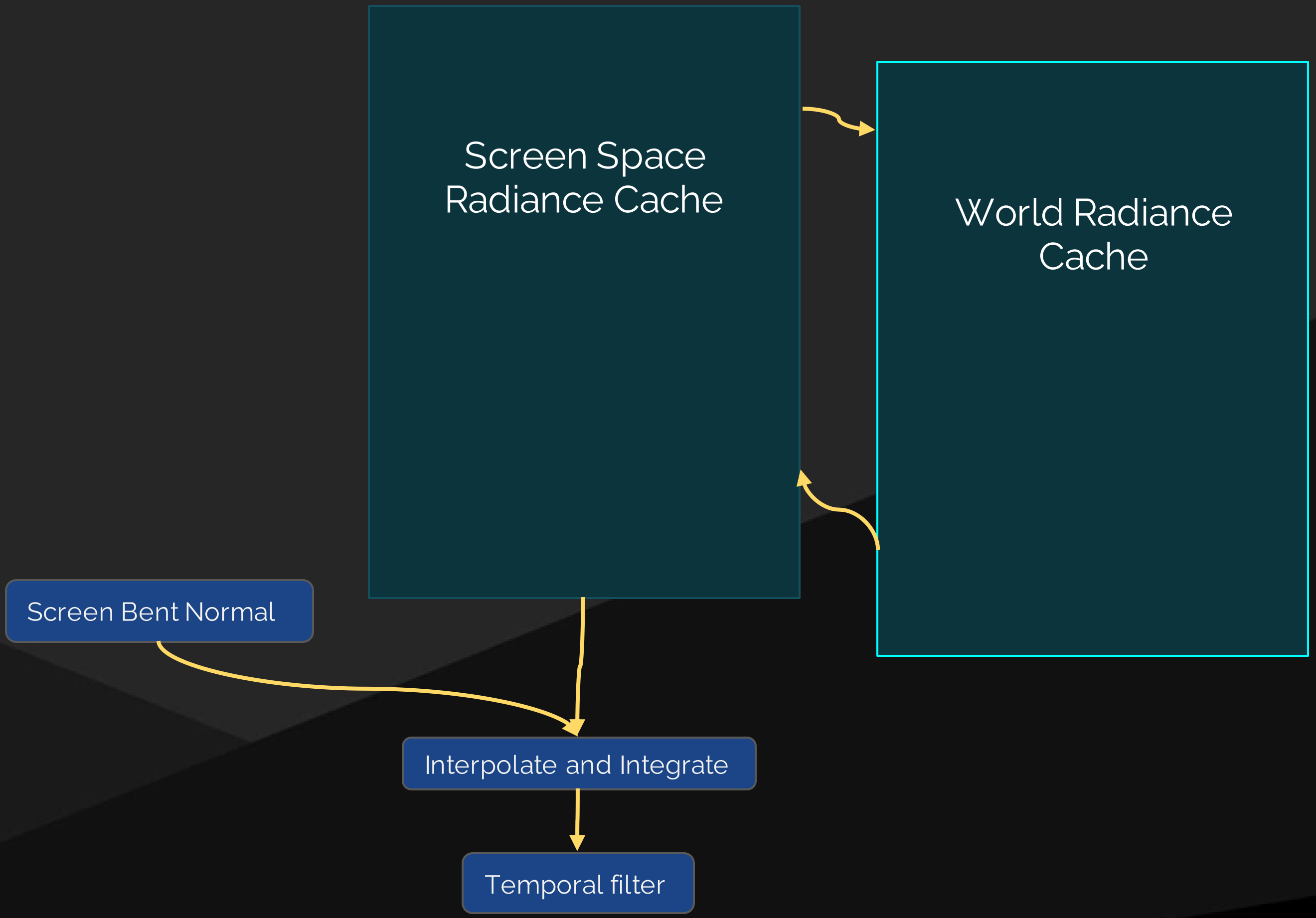
Problem: distant lighting

- Noise from small bright feature increases with distance
- Long incoherent traces are slow
- Distant lighting is changing slowly - opportunity to cache
    - Redundant operations for nearby Screen Probes

Solution: separate sampling for distant Radiance

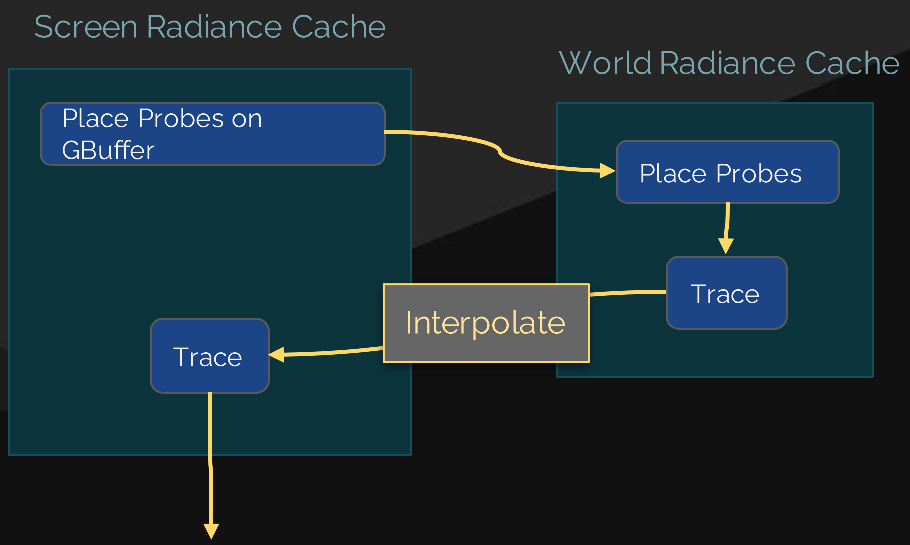
- World space Radiance Caching for distant lighting
  - The Technology of The Tomorrow Children [McLaren 2015]
- Stable error since world space - easy to hide
  - Just like Volumetric Lightmaps



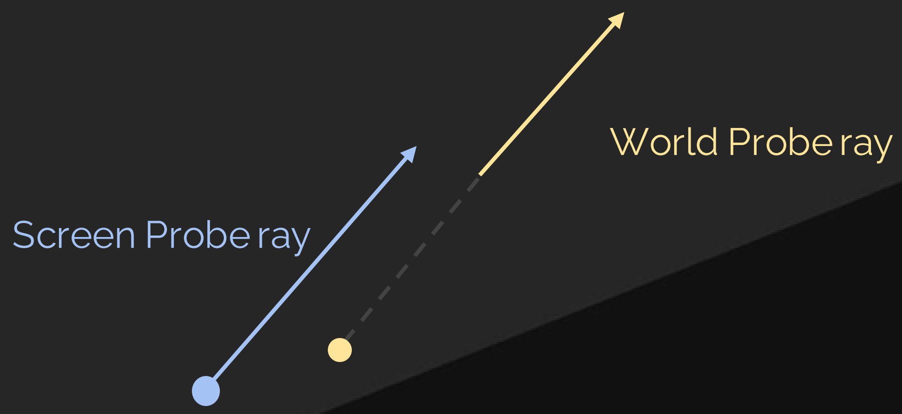

Screen Space Radiance Cache
World Radiance Cache
Screen Bent Normal
Interpolate and Integrate
Temporal filter

Pipeline integration

- Place around Screen Probes
- Then trace to compute Radiance
- Interpolate to solve distant lighting for Screen Probe rays

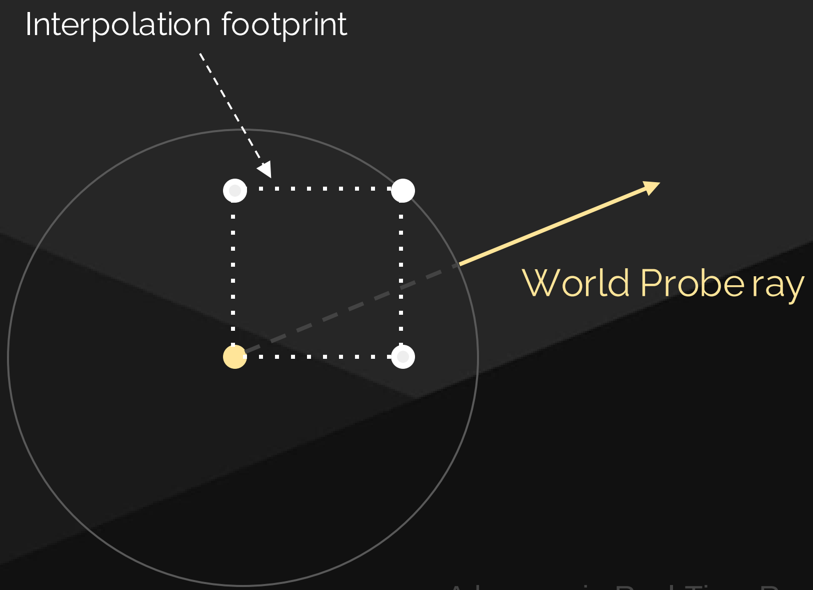


Connecting rays



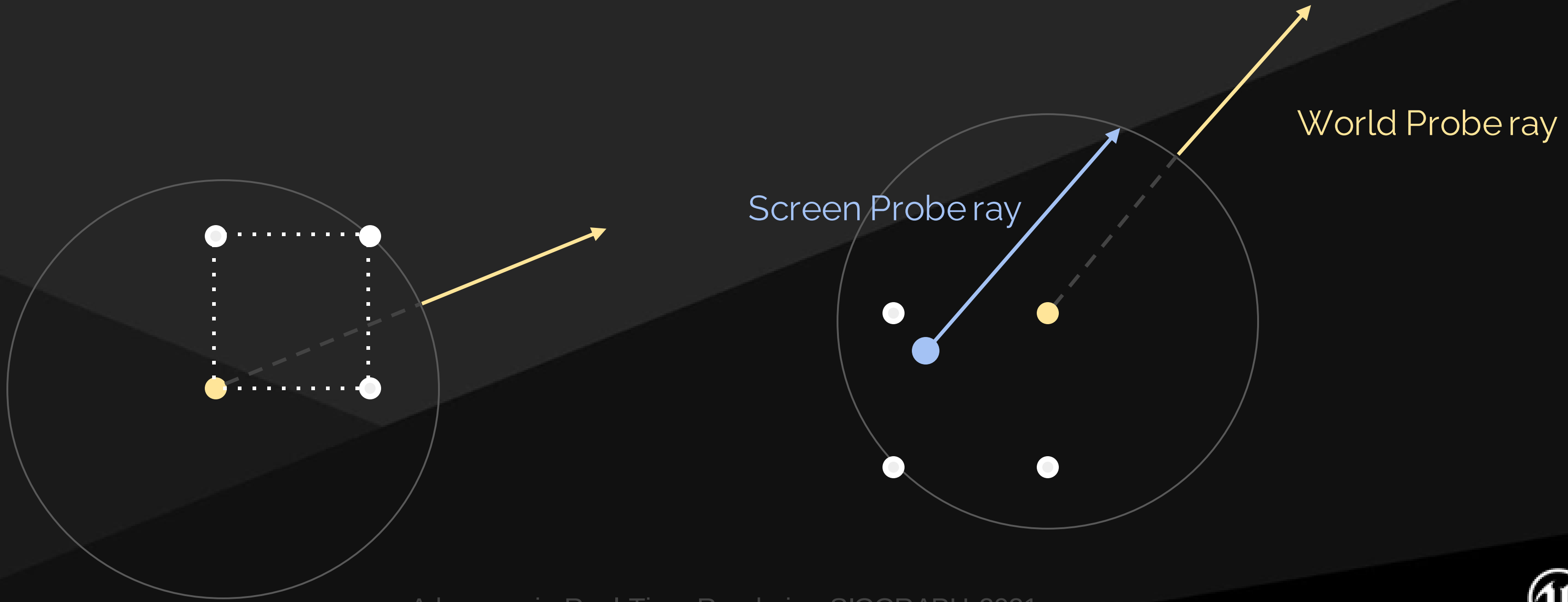
Avoiding self-lighting

- World Probe ray must skip the interpolation footprint



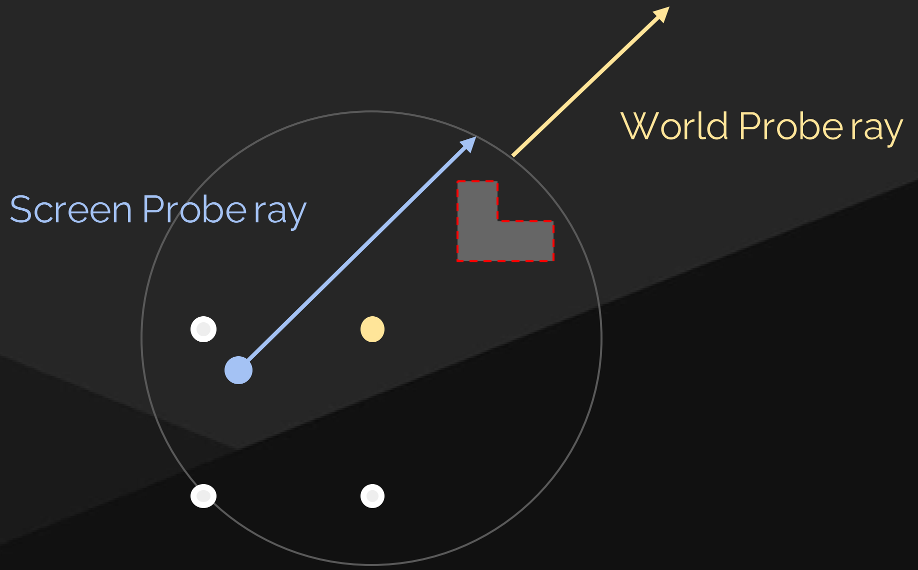
Connecting rays

- Screen Probe ray must cover interpolation footprint + skipped distance



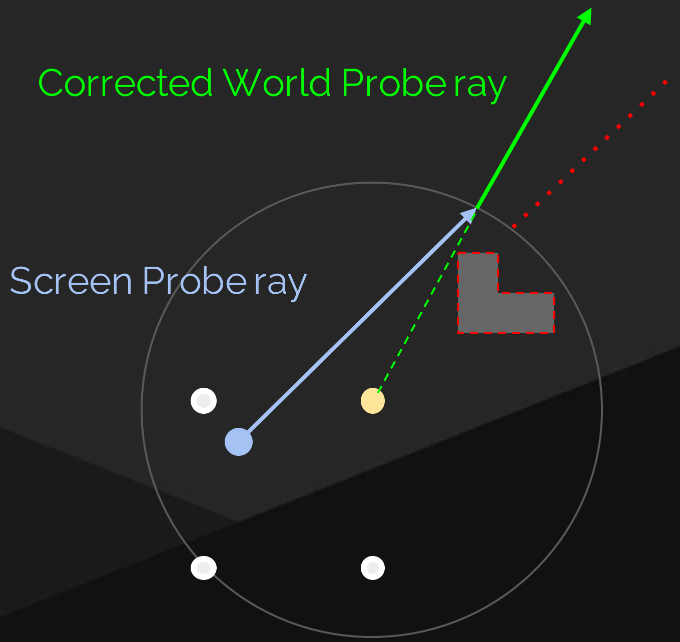
Problem: leaking!

- World probe radiance should have been occluded
  - But wasn't due to incorrect parallax



Solution: simple sphere parallax

- Reproject Screen Probe ray intersection with World Probe sphere

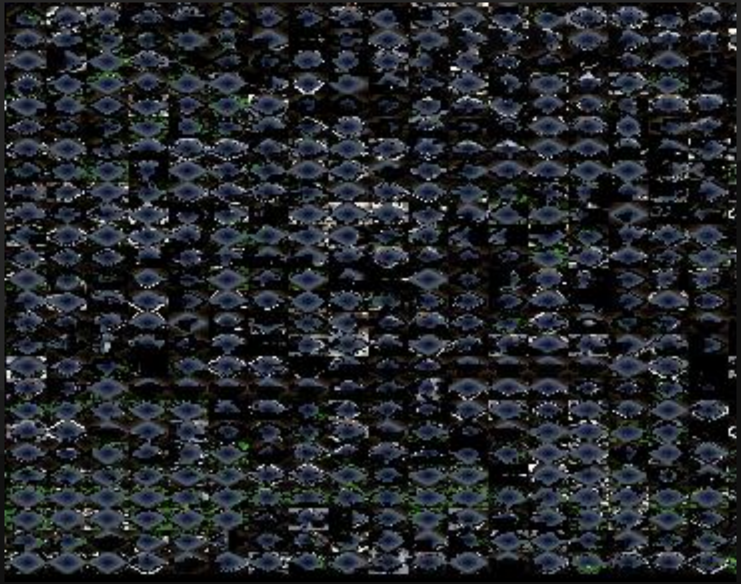


Sparse coverage

- 3d clipmap grids centered around camera storing ProbeIndex into atlas
  - Clipmap distribution maintains bounded screen size

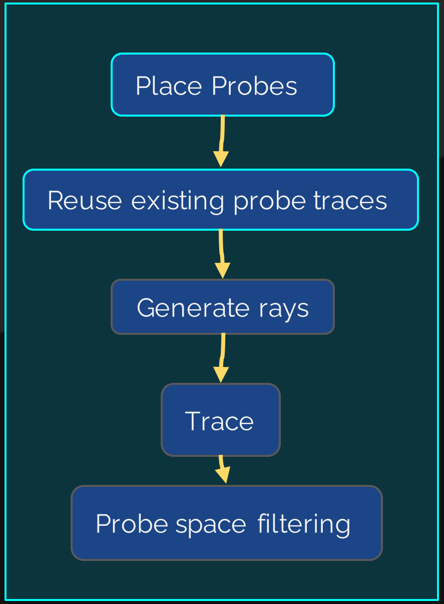
Atlas

- Octahedral probe atlas storing Radiance, TraceDistance
  - Typically 32x32 Radiance per probe



Placement and caching

- Mark any position that we will interpolate from later in clipmap indirections
- For each marked world probe:
    - Reuse traces from last frame, or allocate new probe index
    - Re-trace a subset of cache hits to propagate lighting changes



Problem: highly variable costs

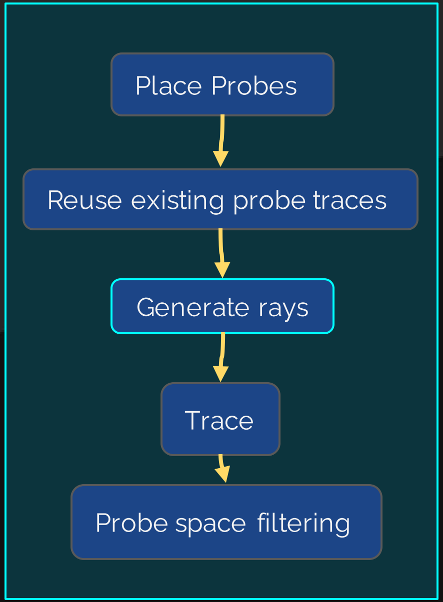
- Fast camera movements and disocclusions require many uncached probes to be traced

Problem: highly variable costs

- Fast camera movements and disocclusions require many uncached probes to be traced
- Solution: Fixed budget for full resolution probes
  - Additional probe traces for cache misses are at lower resolution
  - Additional probe traces for lighting updates are skipped

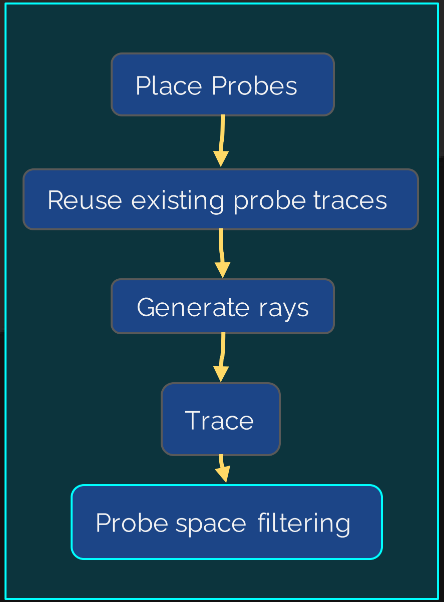
Importance Sampling

- BRDF importance sampling
  - Accumulate BRDF from Screen Probes
  - Dice probes into trace tiles
  - Generate trace tile resolution according to BRDF
- Supersample near camera
  - Up to 64x64 effective resolution
    - 4096 traces!
    - Very stable distant lighting



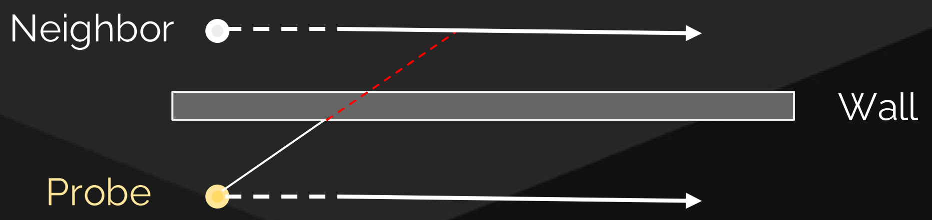
Spatial filtering between probes

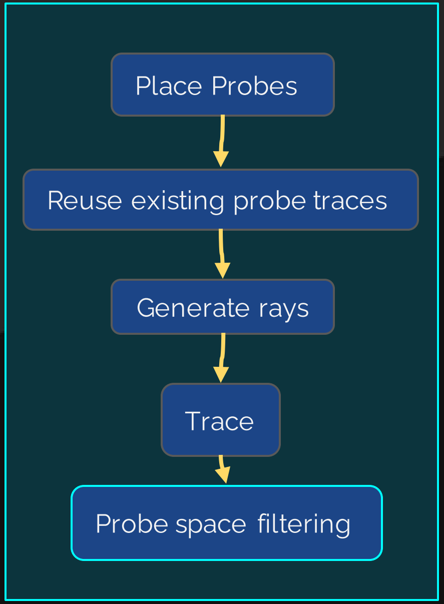
- Reproject neighbor hits again
- Problem: can't assume mutual visibility



Preventing leaking

- Ideally re-trace neighbor ray path through probe depths
- Single occlusion test works well
  - Nearly free - reuses probe depths





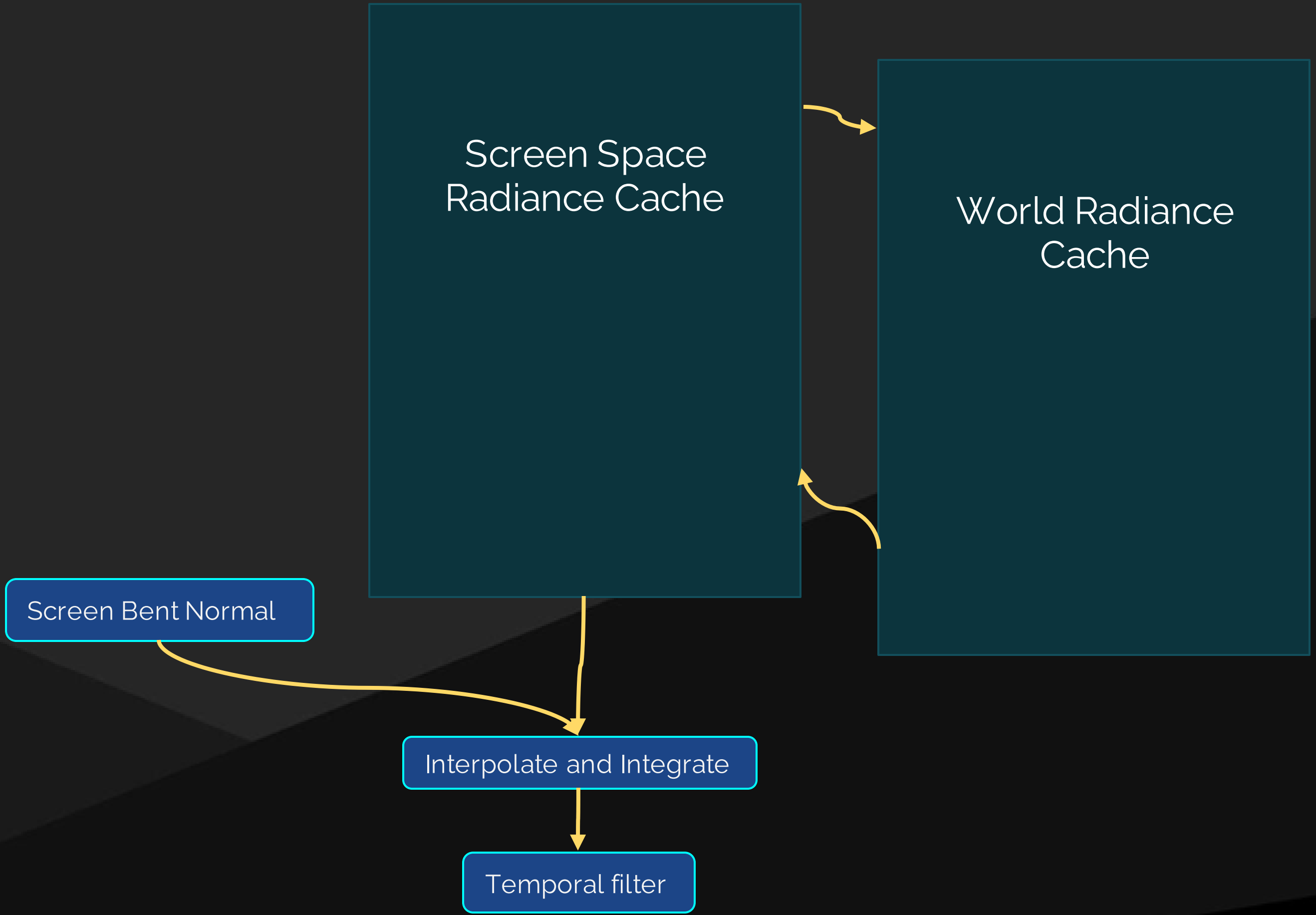
Results

Temporal stability improved

Also used by

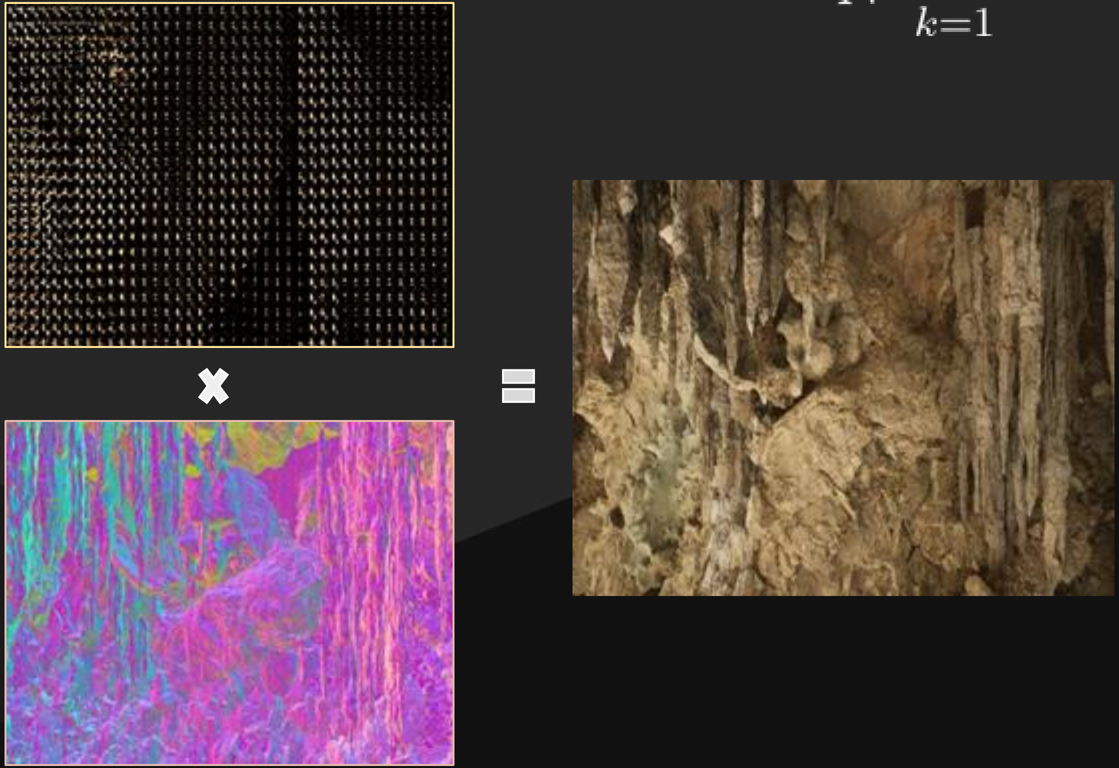
- Guiding Screen Probe importance sampling
- Hair
- Translucency
- Multi-bounce

Full resolution steps


Screen Space Radiance Cache
World Radiance Cache
Screen Bent Normal
Interpolate and Integrate
Temporal filter

Final integration

$$\frac{1}{N}\sum_{k=1}^{N}\frac{\boxed{Li(l)}\,\boxed{fs(l \to v)cos(\Theta l)}}{Pk}$$



Monte Carlo integration noise

- Importance sampling BRDF causes incoherent fetches
  - 8spp * 4 neighbor probe direction lookups
- Can use mips (Filtered Importance Sampling), but that causes self-lighting [Colbert et al 2007]
  - Especially around areas of direct lighting

Convert Probe Radiance to 3rd order Spherical Harmonic

- SH is calculated per Screen Probe
- Full res pixels load SH coherently
- SH Diffuse integration cheap and high quality [Ramamoorthi 2001]

Rough specular

- Ray Traced Reflections expensive at high roughness
  - Converges on diffuse

Rough specular - reuse Screen Probes

- Generate directions from GGX, sample probe radiance
- Automatically leverage probe sampling and filtering already done!

Downsampled tracing loses contact shadows

Full resolution Bent Normal

- Computed with fast screen traces
  - Trace distance coupled to the distance between Screen Probes: ~16 pixels

Integrating with Screen Space Radiance Cache

- Treat Screen Probe GI as Far-Field Irradiance
- Full resolution Bent Normal represents amount of Near-Field
  - Horizon-Based Indirect Lighting [Mayaux 2018]
  - Multi-bounce approximation gives Near-Field Irradiance

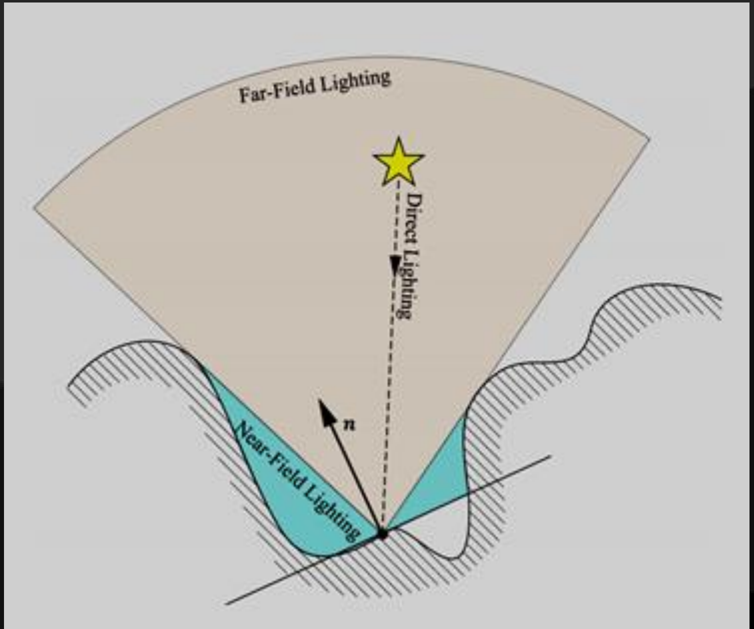


Image credit: Benoit Mayaux

Results

Results

Temporal filter

- Jittering probe position requires reliable temporal filter
- Using depth rejection
  - Stable results, but also slow reaction to lighting changes

Track hit velocity along with hit depth during tracing

Projected area belonging to fast moving objects

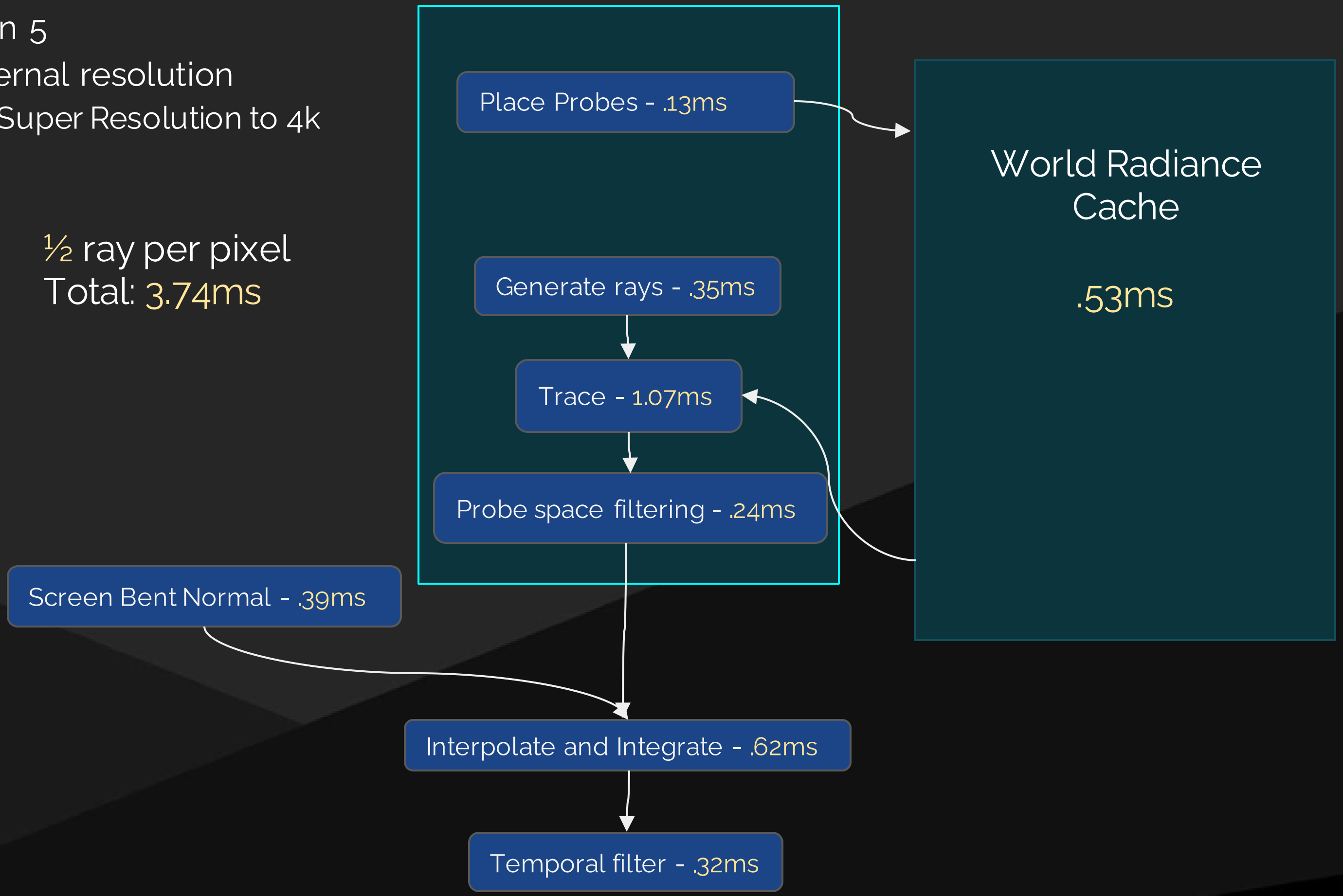
Switch to fast update mode when traces hit fast moving object

- Lower temporal filter, raise spatial filter

Final Gather performance

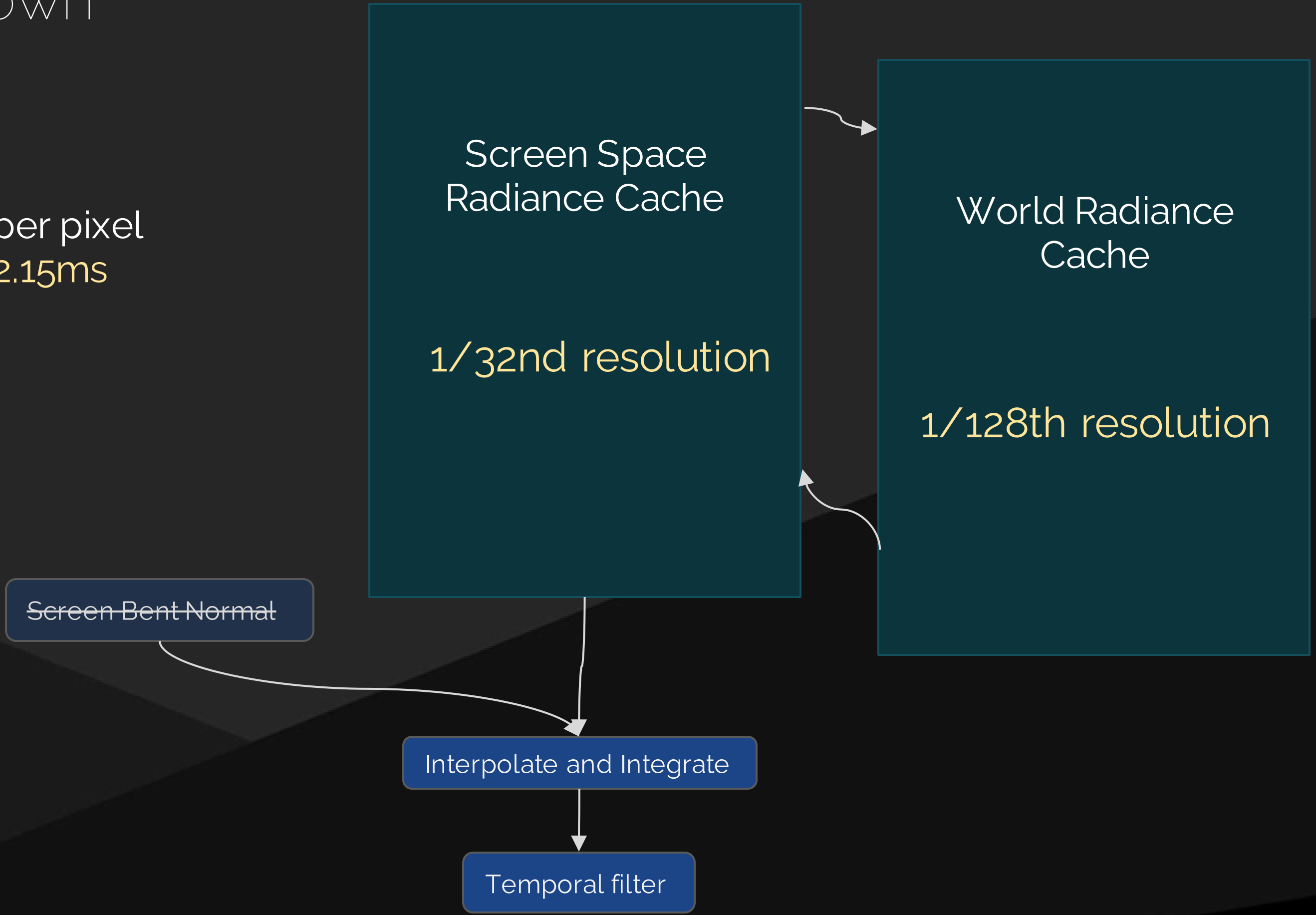
Playstation 5
1080p internal resolution
Temporal Super Resolution to 4k

½ ray per pixel
Total: 3.74ms



Scaling down

⅛ ray per pixel
Total: 2.15ms




⅛ ray per pixel
2.15ms Final Gather


½ ray per pixel
3.74ms Final Gather


EPIC


2 rays per pixel
4.23ms on 2080 TI
Image credit: Rafael Reis


Image credit: Rafael Reis

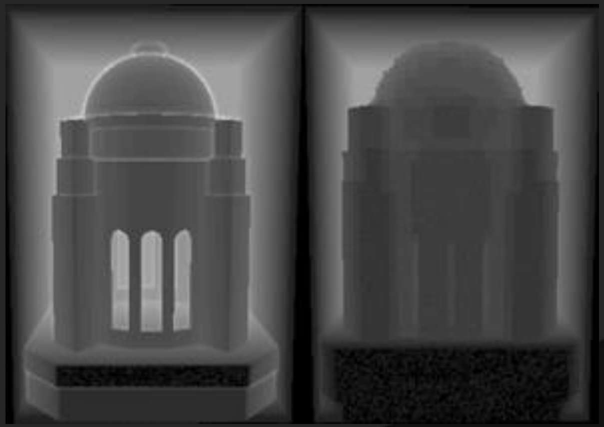
Image credit: Rafael Reis

Provides the Final Gather for Lumen in Unreal Engine 5

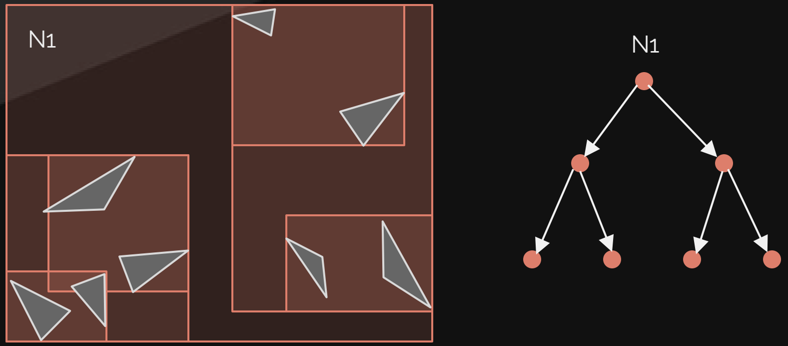
- Dynamic Global Illumination, Shadowed sky lighting, Emissive
- Opaque materials only
  - GI on transparency and fog provided through separate technique
- Rough specular integrates with Lumen's Ray Traced Reflections
- Documentation at docs.unrealengine.com

Supports Lumen's hybrid tracing

- Software Ray Tracing
  - Per-Instance Sparse Signed Distance Fields



- Hardware Ray Tracing
  - Per-Instance Triangle BVH



Future Work

- Disocclusion quality
- Temporal stability in highly dynamic scenes
- Applying Screen Space Radiance Cache to Lumen's Surface Cache for multi-bounce GI

Only presented the opaque Final Gather

- Many other important parts of Lumen!
  - Surface Caching
  - Software Ray Tracing
  - Hardware Ray Tracing
  - Reflections
  - Transparency GI

Lumen Team

Krzysztof Narkowicz

Patrick Kelly

Other Acknowledgements

Unreal Engine Ray Tracing

- Yuriy O'Donnell
- Juan Canada
- Yujiang Wang
- Kenzo ter Elst

Presentation review

- Brian Karis
- Michal Valient

References

- Irradiance Volumes for Games [Tatarchuk 2012]
- Taking Killzone Shadow Fall Image Quality into the Next Generation [Valient 2014]
- Ray-Traced Irradiance Fields [McGuire 2019]
- Spatiotemporal Variance-Guided Filtering: Real-Time Reconstruction for Path-Traced Global Illumination [Schied et al 2017]
- Fast Denoising with Self Stabilizing Recurrent Blurs [Zhdan 2020]
- Practical Global Illumination with Irradiance Caching [Křivánek et al 2007]
- Structured Importance Sampling of Environment Maps [Agarwal et al 2003]
- The Technology of The Tomorrow Children [McLaren 2015]
- An Efficient Representation for Irradiance Environment Maps [Ramamoorthi 2001]
- Horizon-Based Indirect Lighting [Mayaux 2018]
- Real-time Shading with Filtered Importance Sampling [Colbert et al 2007]