



MEET LIGHTSPEED STUDIOS AT GDC2023

March 20-24, 2023 | San Francisco, CA

HSGL:

Cross-Platform Hierarchical Surfel Global Illumination

Production Team: **Jiancong HUANG**, Minmin GONG, **Huiwen JIANG**, **Jiaxuan ZHANG**,
Junfeng LI, Wei LAN, Yucong, PAN, Kai DAI, Jiaqi GUO, Mali LIU, Lihua LI

LIGHTSPEED STUDIOS

March 20-24, 2023 | San Francisco, CA

HSGI

Cross-Platform Hierarchical Surfel Global Illumination







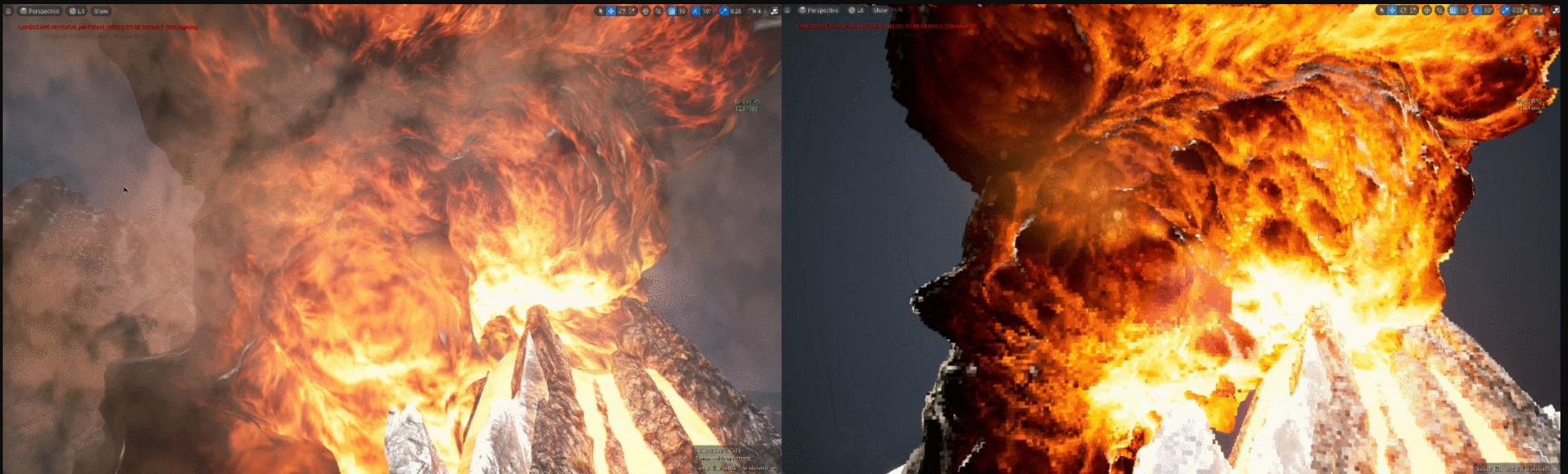
Motivation

- Reason

- Huge Open World
- Dynamic Geometry
- Dynamic Lighting

- Target

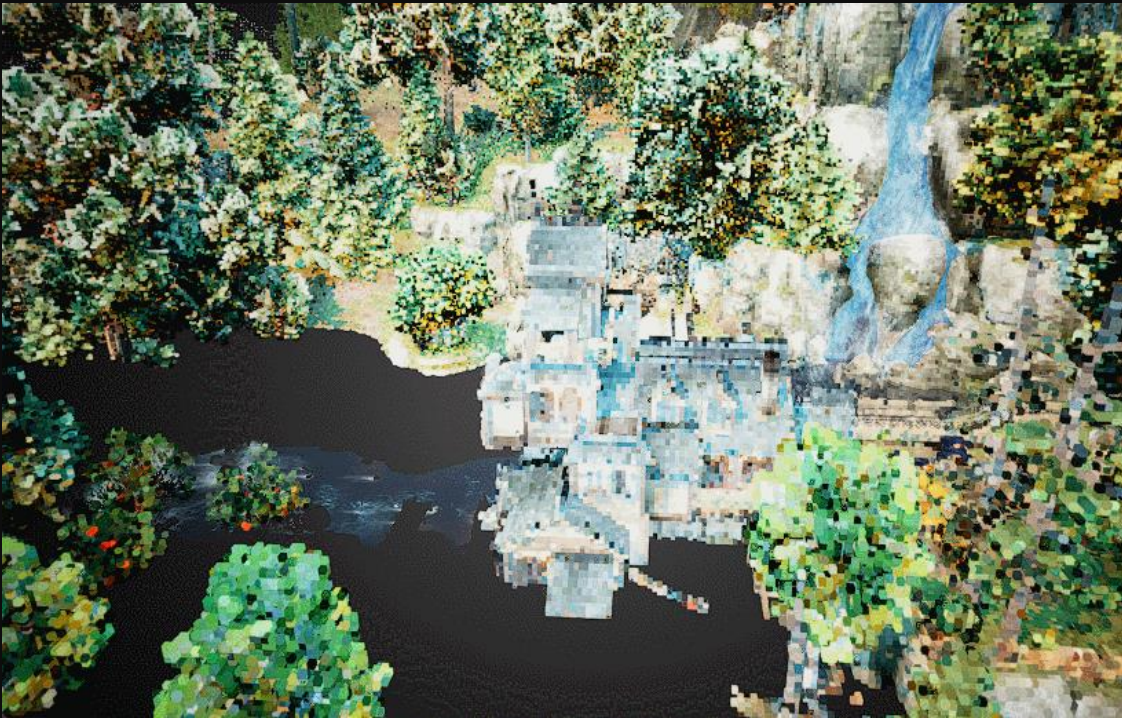
- PC/Console
 - comparable result with path tracing
- Mobile
 - controllable performance



Idea

- **Problem 1. Scene Description?**

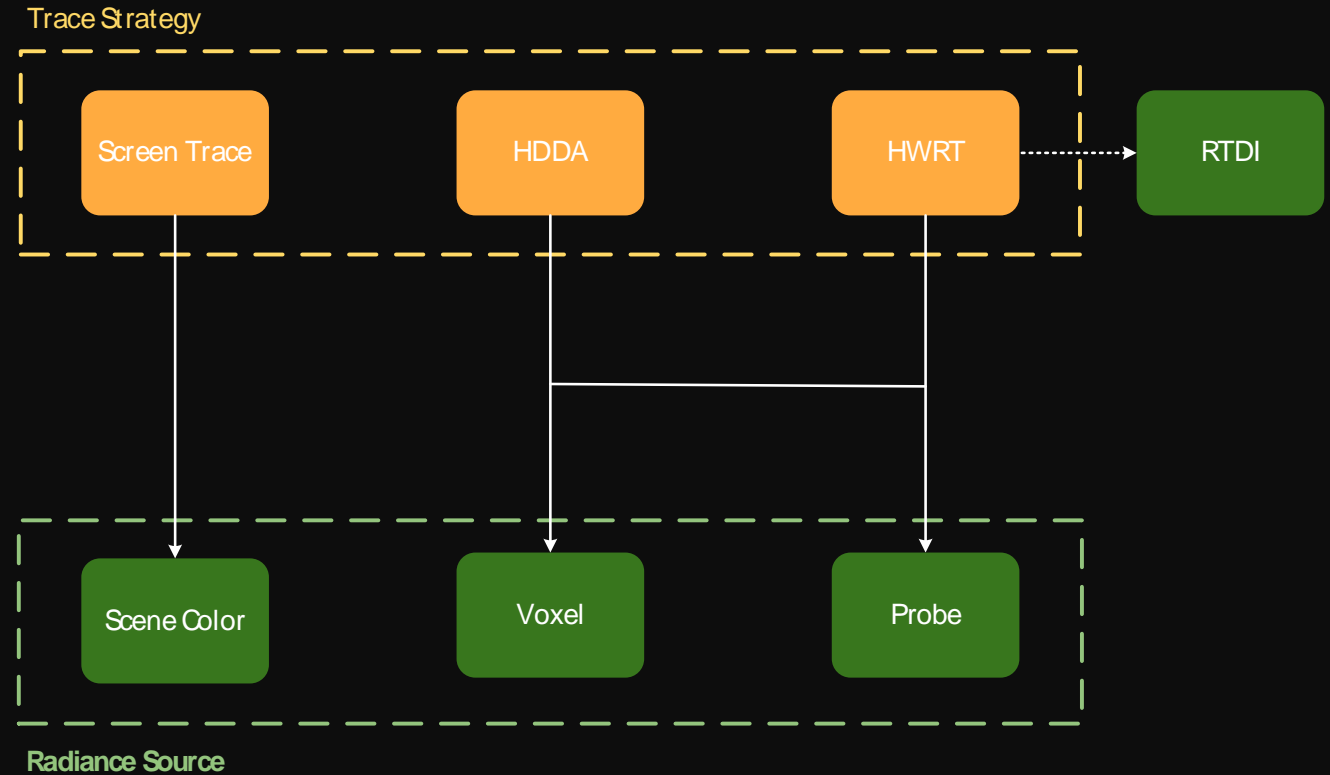
- Update dynamic geometry per frame at a large scale
- Inspired by 3D reconstruction with RGB-D
- Surfel = Voxel + Normal



[BundleFusion, 2017]

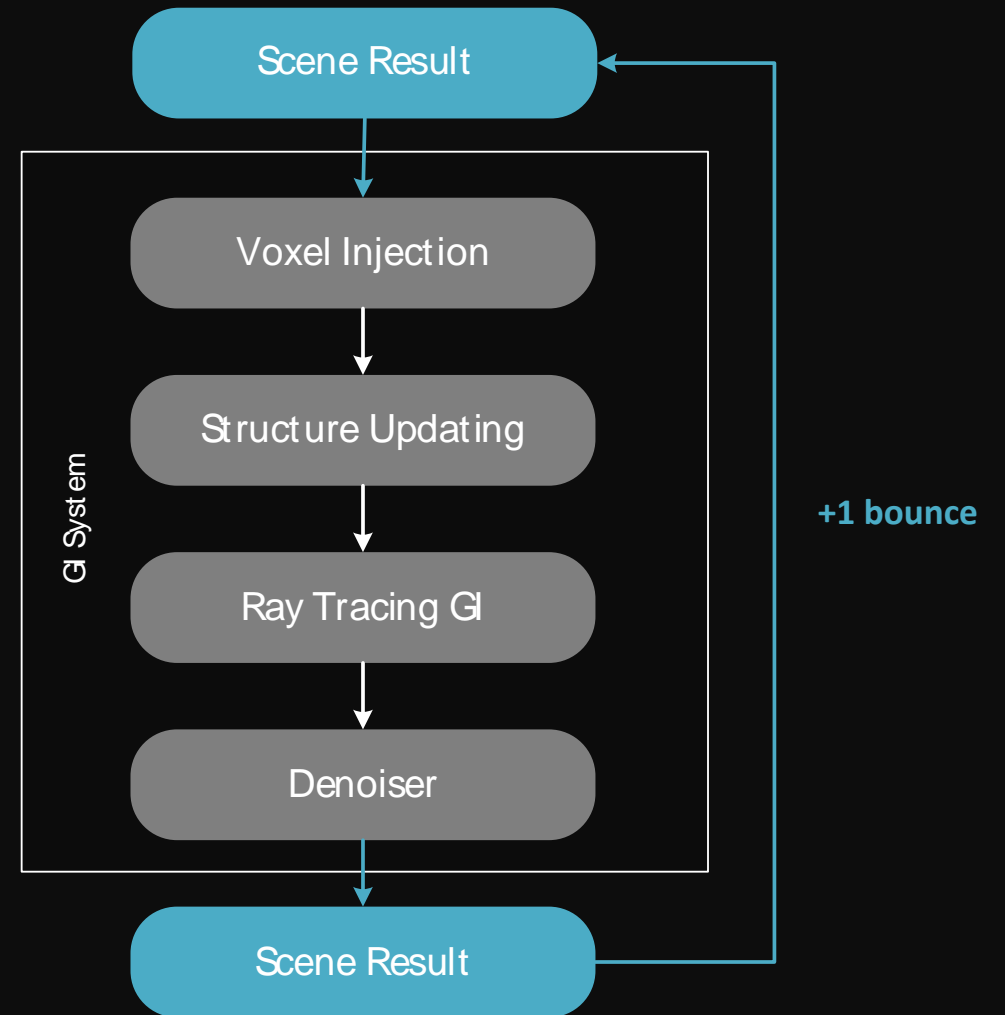
Idea

- Problem 1. Scene Description?
- Problem 2. Trace Ray?
 - PC/Console: HWRT/HDDA/Screen Trace
 - Mobile: HDDA



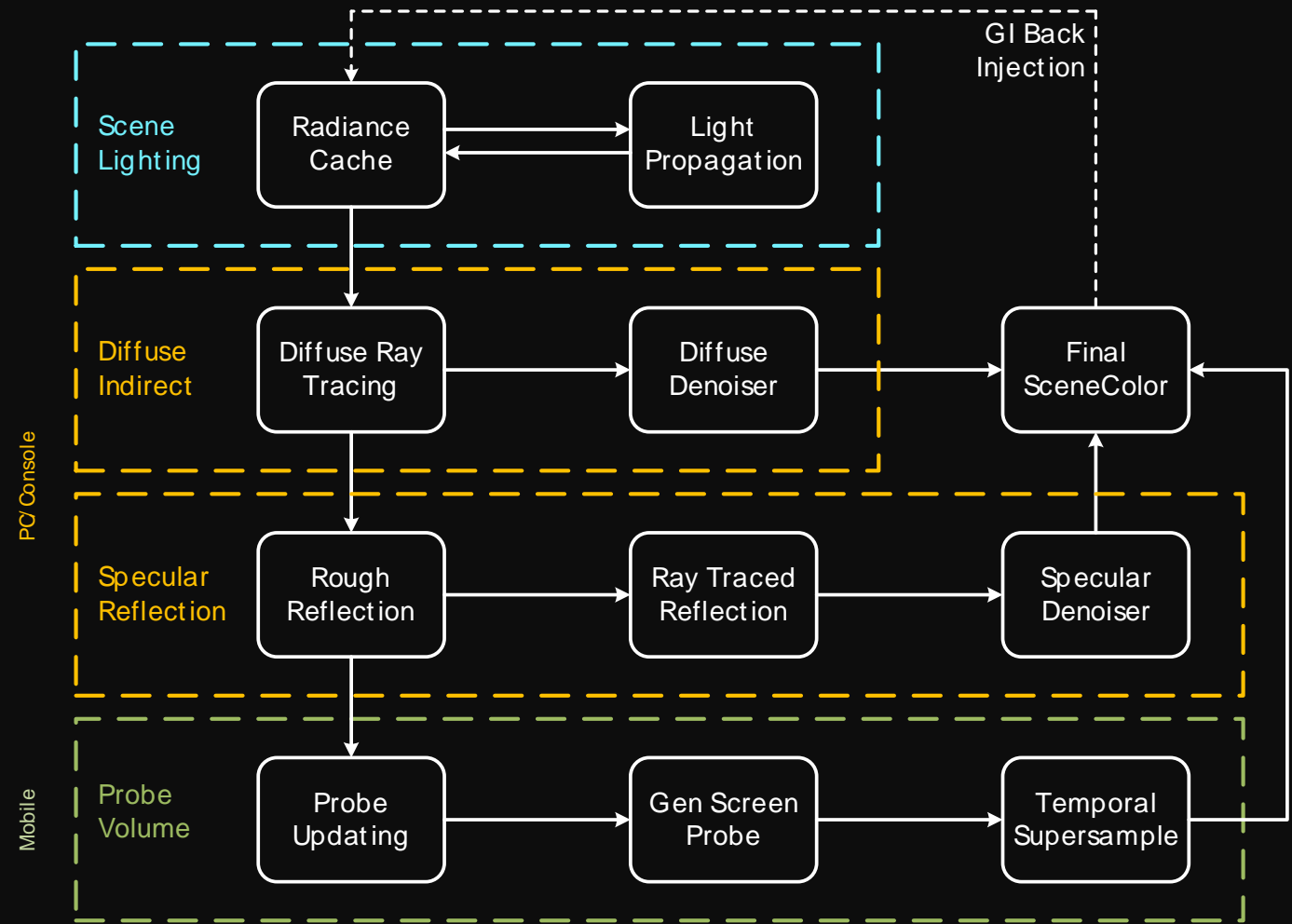
Idea

- Problem 1. Scene Description?
- Problem 2. Trace Ray?
- Problem 3. Multi-bounce?
 - Radiance cache feedback
 - Voxel light propagation



Idea

- Problem 1. Scene Description?
- Problem 2. Trace Ray?
- Problem 3. Multi-bounce?
- Problem 4. Denoise?
 - PC/Console: ReSTIR
 - Mobile: Screen Probe



Outline

- **Radiance Cache Structure**

- Hierarchical Structure
- Injection
- Voxel Tracing

- **Mobile Solution**

- Probe Volume
- Deferred Screen Probe
- Final Gather

- **PC/Console Solution**

- Ray Tracing
- Diffuse GI
- Reflection



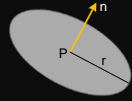
Radiance Cache



Hierarchical Structure

- **Radiance Cache:**

- Surfel = Voxel + Normal



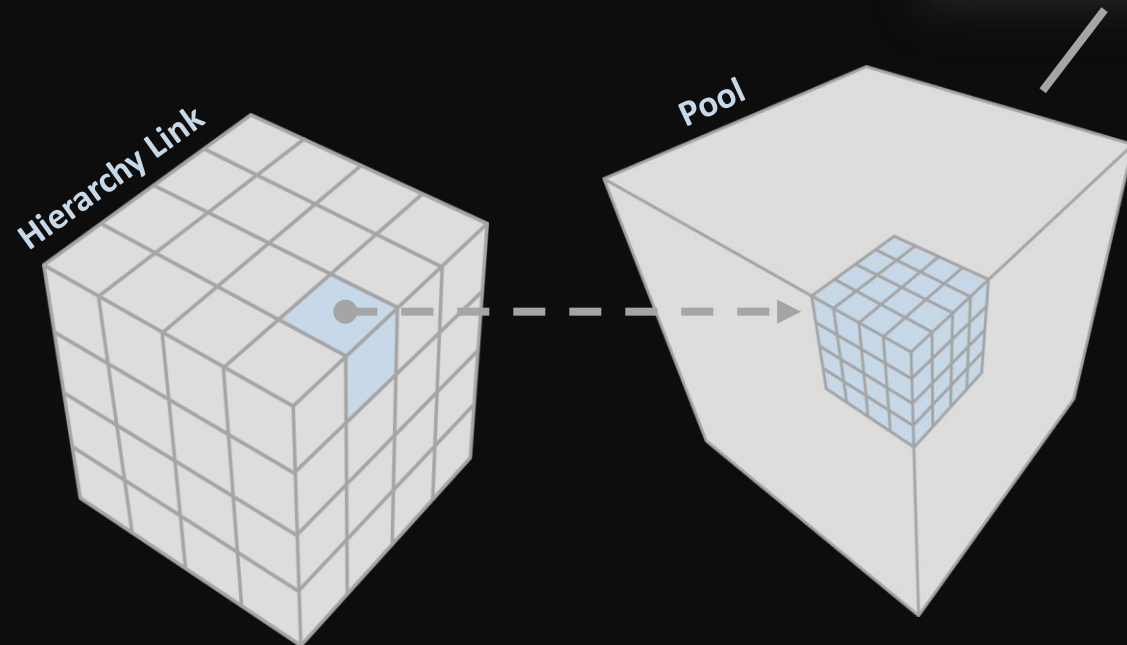
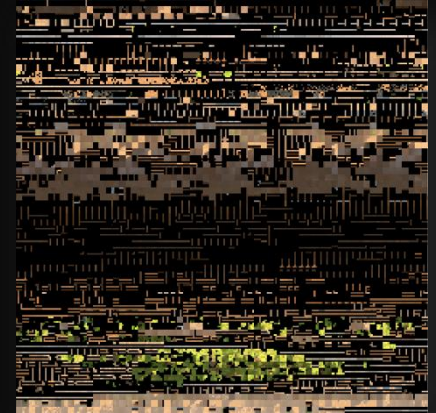
- **1 brick = 4x4x4 voxels**

- **Clipmapping**

- Sparse Volume Texture

- **Resources:**

- Brick ID:
 - Physical memory offset
 - Just like Page ID of virtual texture
- Memory Pool:
 - Indexed by Brick ID
 - Contains voxel data
- Hierarchy Link:
 - Dense Brick ID structure
 - Content:
 - If Allocated: Brick ID
 - Else: zero
- Allocator: tracking unused Brick ID








Voxel Data

- **PC/Console:**

- 4 Textures
- 128 bits

- **Mobile:**

- 2 Textures
- 64 bits

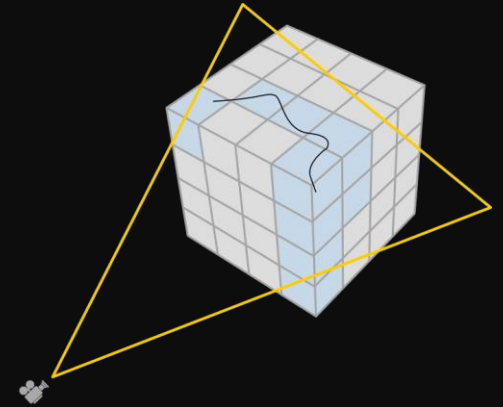
Term	Format	bits
Diffuse Color	r8b8g8	24 bits
Direct Outgoing Radiance	r11b11g10	32 bits
 Indirect Outgoing Radiance	r13g13b12	38 bits
 Normal	r8b8	16 bits
 Alpha	r8	8 bits
Age Weight	r8	8 bits
 Masks	uint	2 bits
Total (PC/Console)	4 textures	128 bits
 Total (Mobile)	2 textures	64 bits



Injection

- **Screen Injection:**

- Generate one thread per voxel:
 - $\text{sample} < \text{depth}$ \Rightarrow clear
 - $\text{sample} > \text{depth}$ \Rightarrow skip
 - $\text{sample} \approx \text{depth}$ \Rightarrow update (voxel intersects with depth)



Injection

- There're too many threads if we update all voxels!

- **Voxel Volume:**

- Mobile: 100+m
- PC/Console: 500+m

- **Tile-Based Injection**

- **Tile-based AABB Culling**

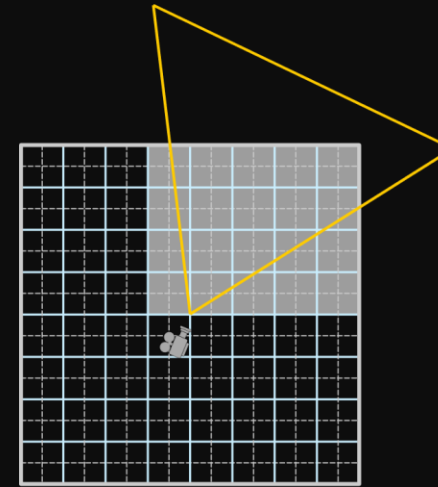
- 1 Tile = 8x8x8 Bricks
- Amortize update rate based on distance

- **Brick Collection**

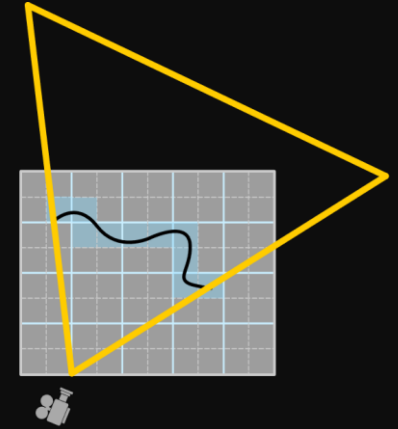
- 1 Brick = 4x4x4 Voxels
- 8-vertices test

- **Voxel Injection**

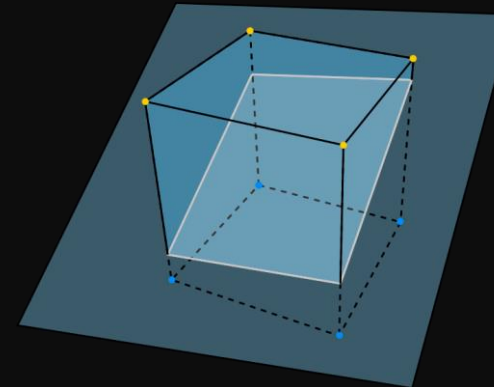
- Sample 3D point in voxel
- Get scene color/depth from the render pipeline



(a) Tile-based AABB Culling



(b) Brick Collection



(c) 8-Vertices Test



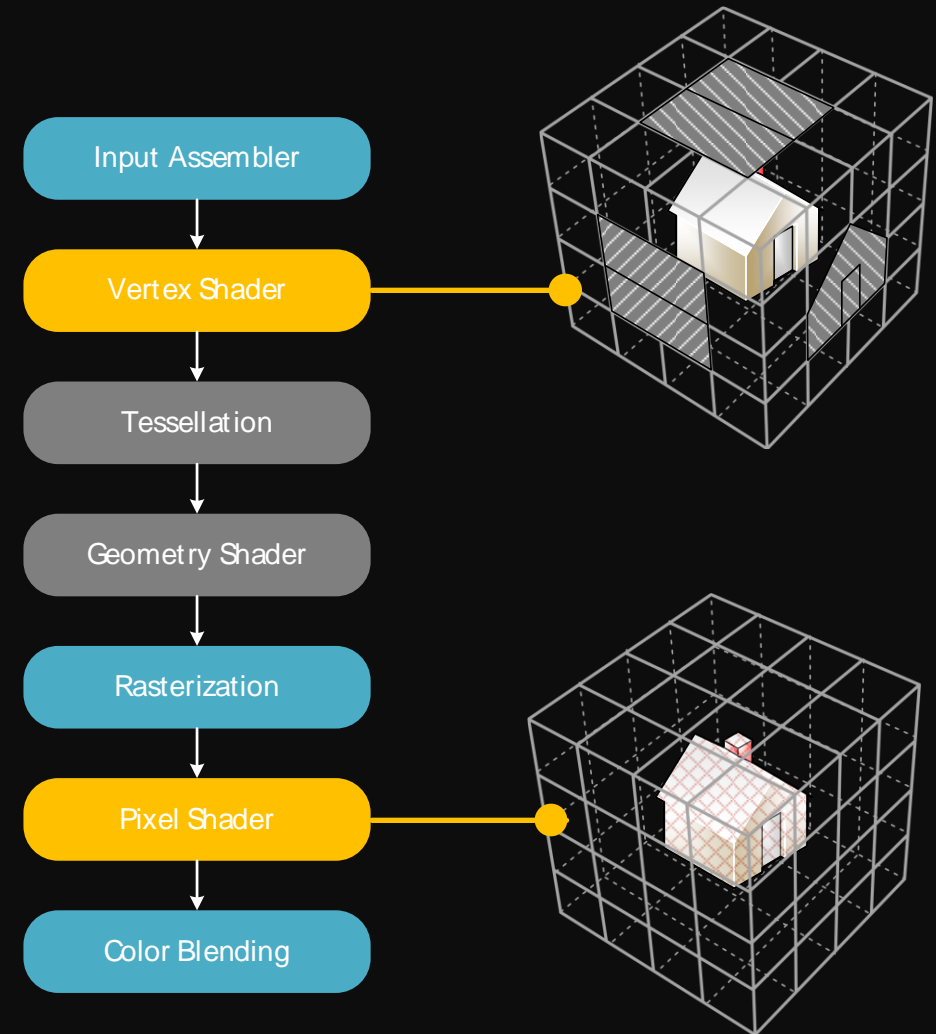
(d) Voxel Inject

Injection

- What about the objects outside the frustum?

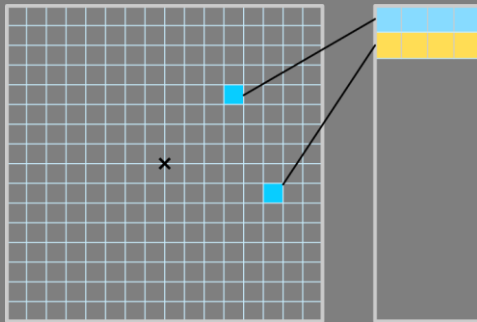
- **Per-Object Voxelization:**

- Disable depth test
- Orthographic projection
- 3 axis: instance draw
- Material shader
- (optional) forward lighting

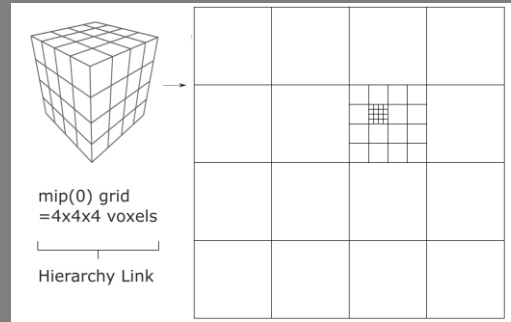


Software Ray Trace

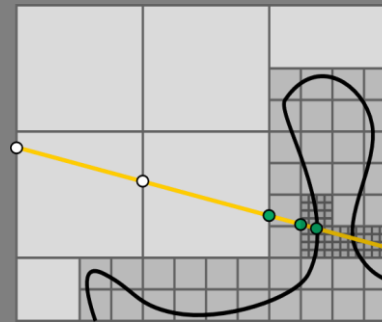
- **Voxel Trace** [GVDB 2016]
 - HDDA
 - Compact hierarchical voxel occupy data
 - 1 bit per voxel occupancy
 - uint64 per fetch
 - sparse mip 0, trace performance for memory
 - Optimize I/O cost



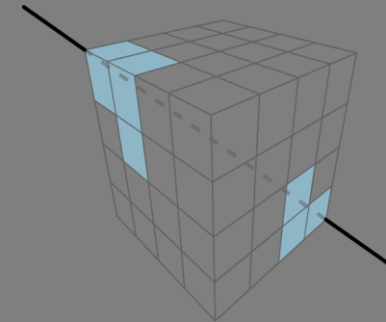
(a) Hierarchy Link



(b) Occupy Structure



(c) Ray Tracing

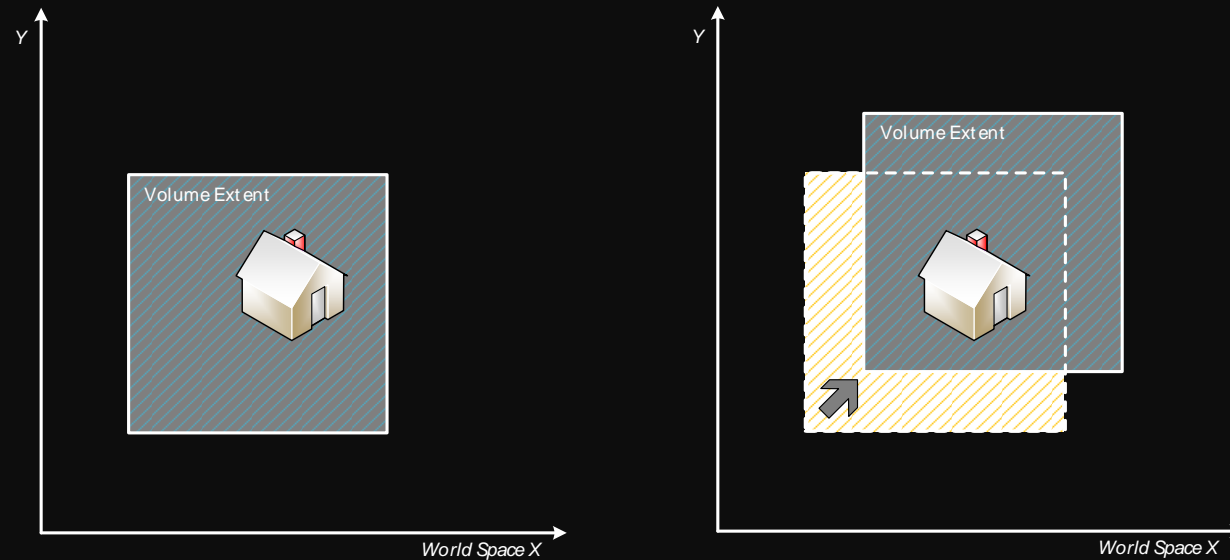


(d) Step DDA



Volume World Shift

- Update HierarchyLink
- Toroidal Addressing
 - Only the **yellow zone** needs to be clear
 - Just mark a clear flag per voxel/brick



Mobile Solution



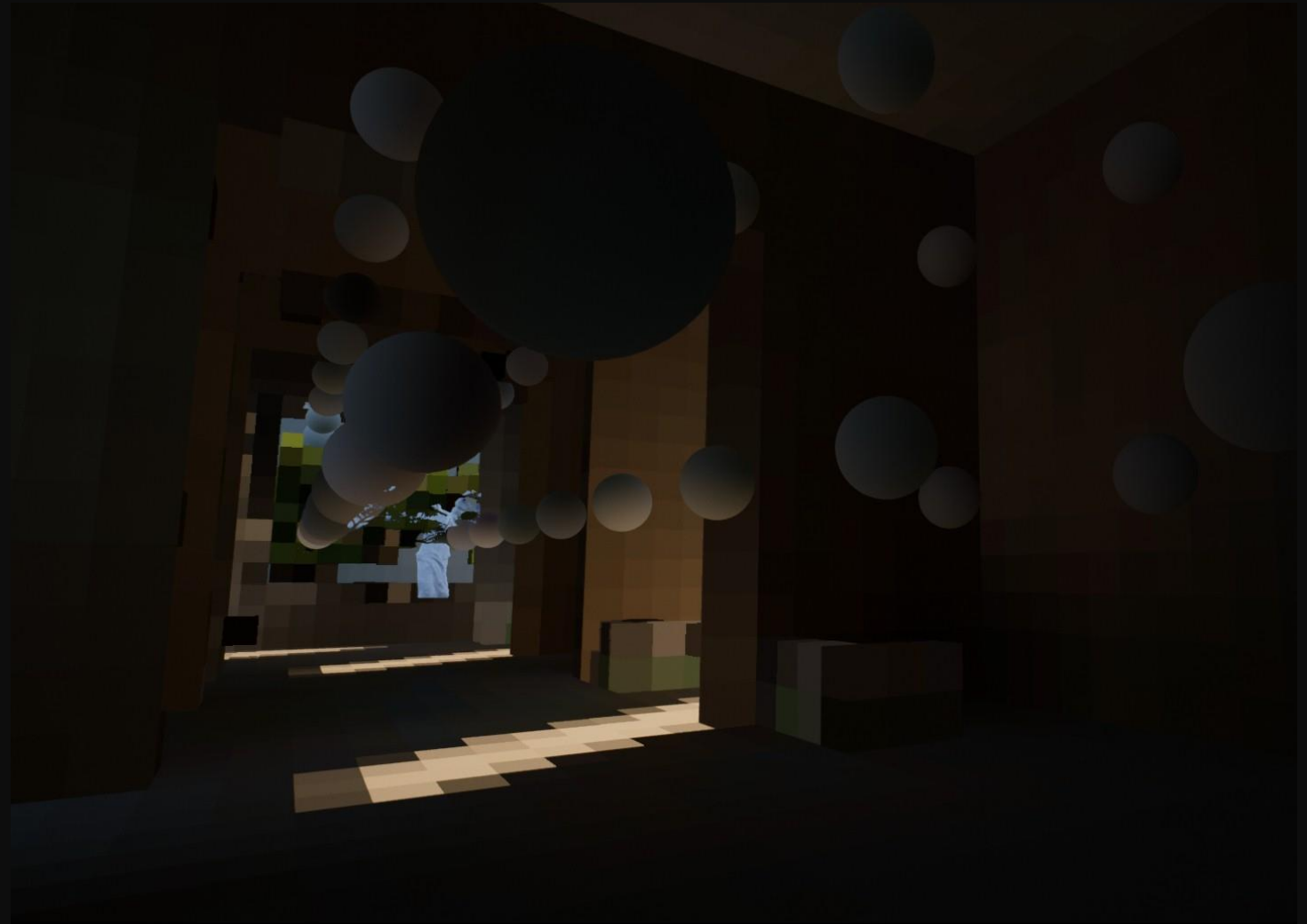
Idea

- Challenge – Performance!

- 1 spp ray trace ✗
- Probe solution ✓

- Idea :

- Place volumes of probes in the 3D world [DDGI 2019]
- Simplify probe format
- Reduce rays



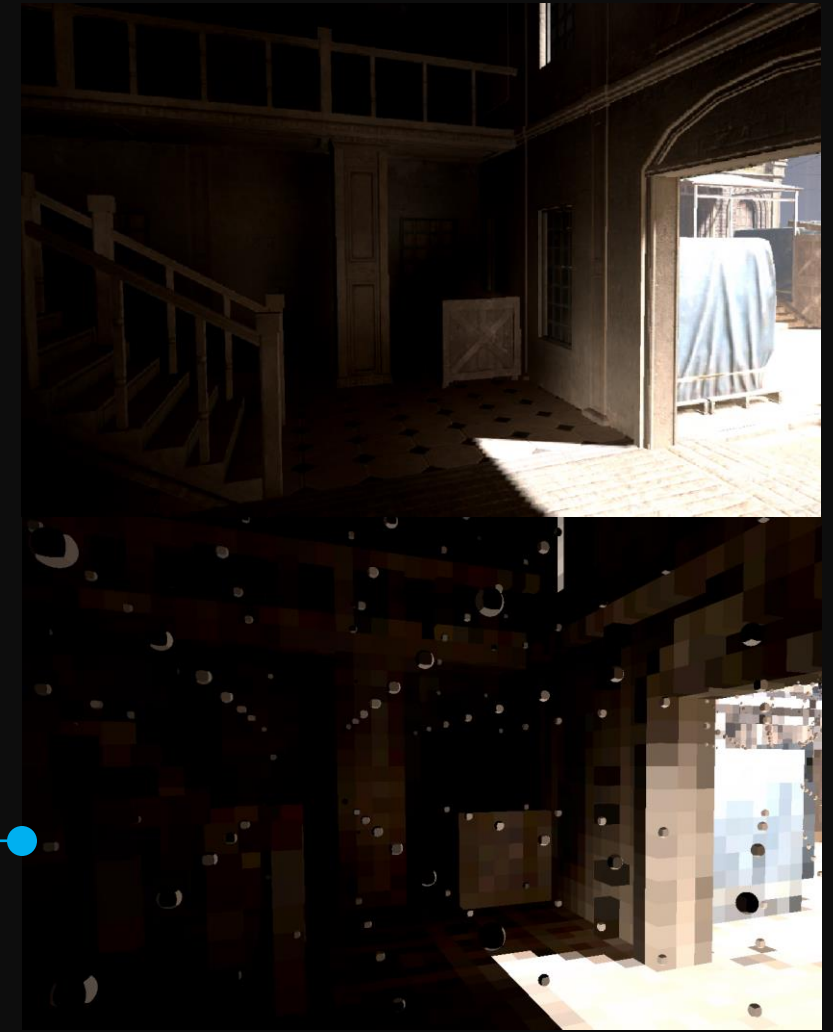
Probe Volume

- **Format**

- **Ambient Cube** – irradiance and age
- Visibility – VSM
- Relocation Info – index and alpha

- **Update**

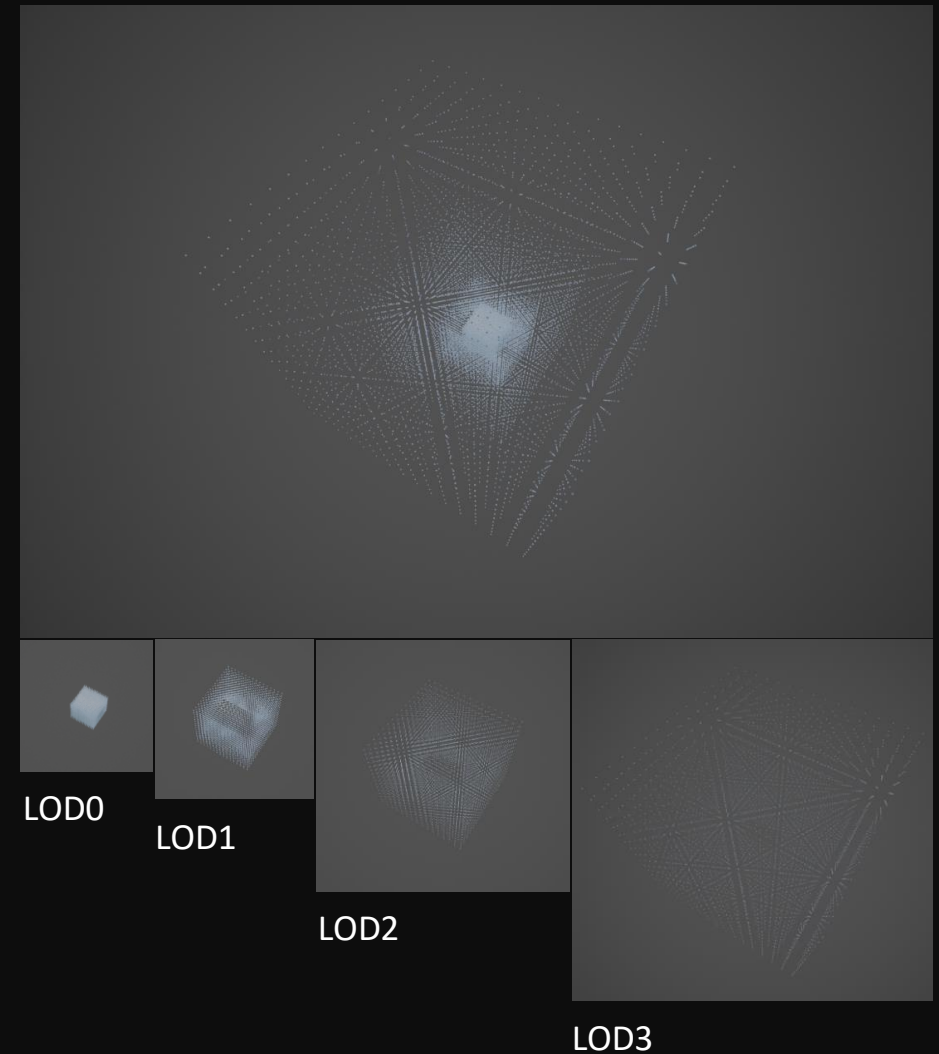
- Collect probes close to the surface – with voxel hierarchy structure
- Each frame rand 1 face to update – trace 1 ray
- Monte Carlo integration
 - $\int_{\Omega} f_r(x, \omega_i, \omega_o) L_i(x, \omega_i) \langle \omega_i \cdot n \rangle d\omega_i$
 - $= \frac{c}{\pi} \int_{\Omega} L_i(x, \omega_i) \langle \omega_i \cdot n \rangle d\omega_i$
 - $= \frac{c}{\pi} \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{N=1}^N \frac{L_i(x, \omega_i) \langle \omega_i \cdot n \rangle}{P_k}, P_k = \frac{\langle \omega_i \cdot n \rangle}{\pi}$
 - $= c \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{N=1}^N L_i(x, \omega_i)$



Probe Volume Structure

- **Clipmap:**

- Support far-field
- More details for near-field
- Variable updating rate for different LODs
- How many?
 - PC/Console: 4 LODs
 - Mobile: 2 LODs

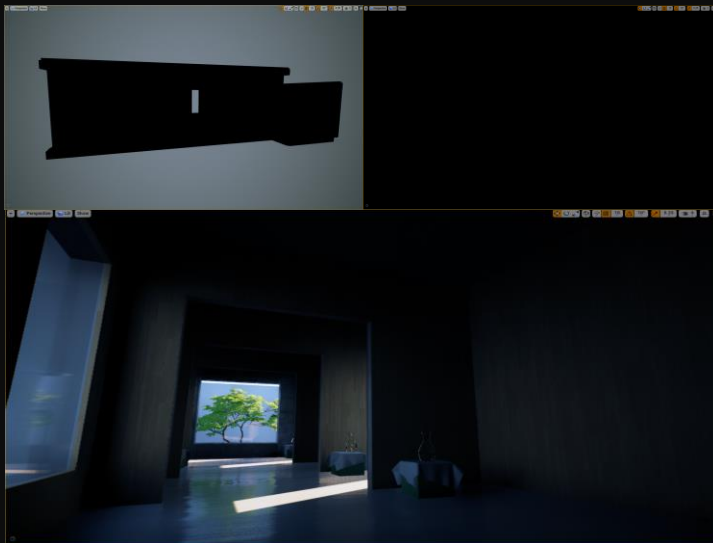


The First Bounce

- Calculate forward lighting in per-object voxelization
 - The first bounce provide most lighting
 - Add fake ambient skylight to initialize voxels



Multi-bounced



Without direct lighting



With fake ambient environment

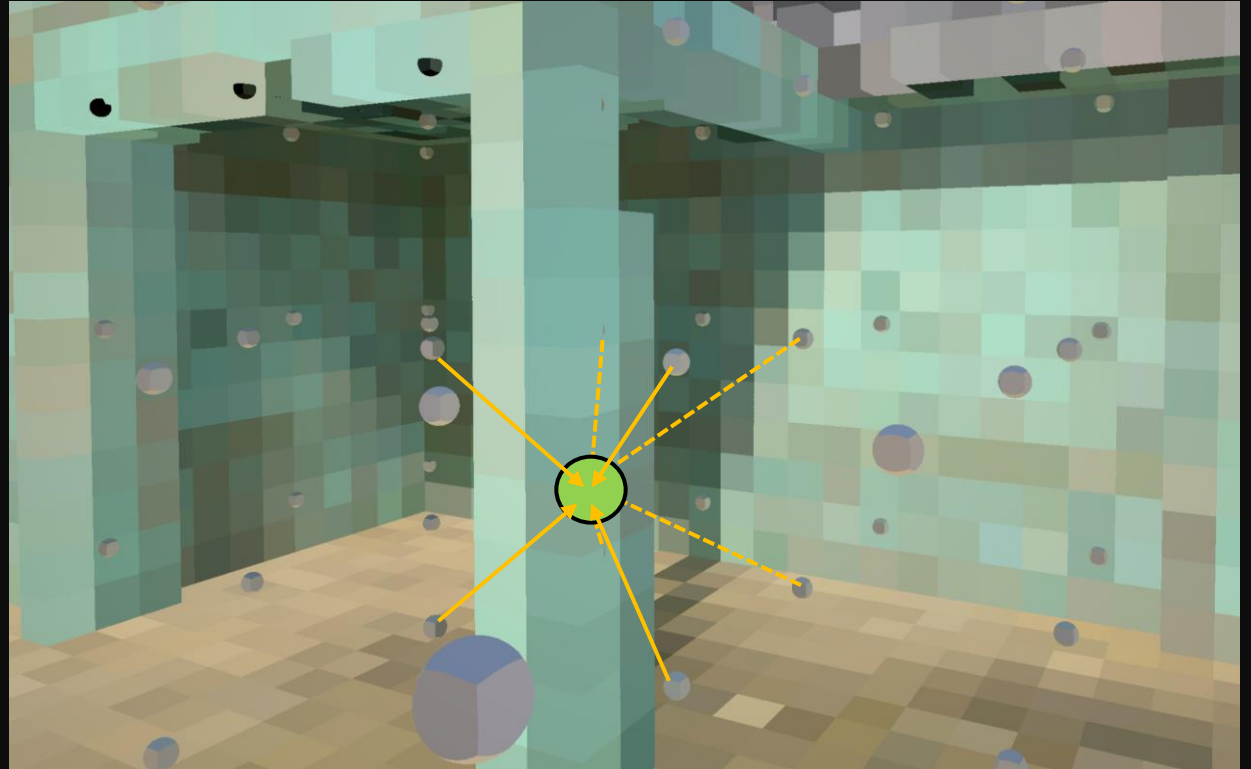


With direct lighting



Problem

- Performance!
- The target device can only reach **30fps**
 - Forward pass gathers irradiance from the 3D probe (ambient cube + VSM) directly
- Analysis
 - Final gather from 8-probe interpolation
 - Too many texture sampling and ALU
 - Most neighboring pixels have repeated samples
 - With slightly different weights
- Idea
 - Deferred Indirect Lighting
 - Screen Probe

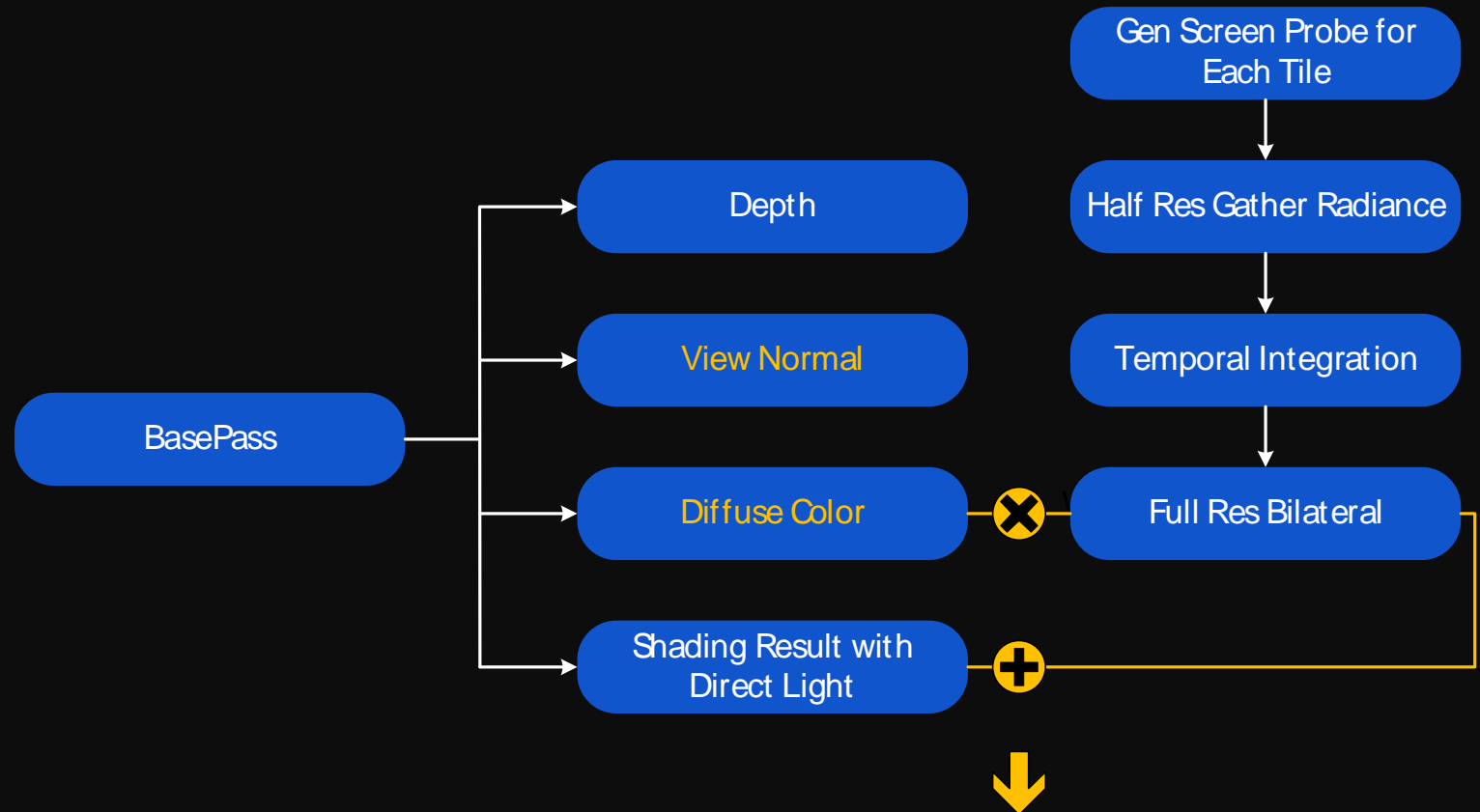


Deferred Screen Probe

- Forward pipeline (optional)

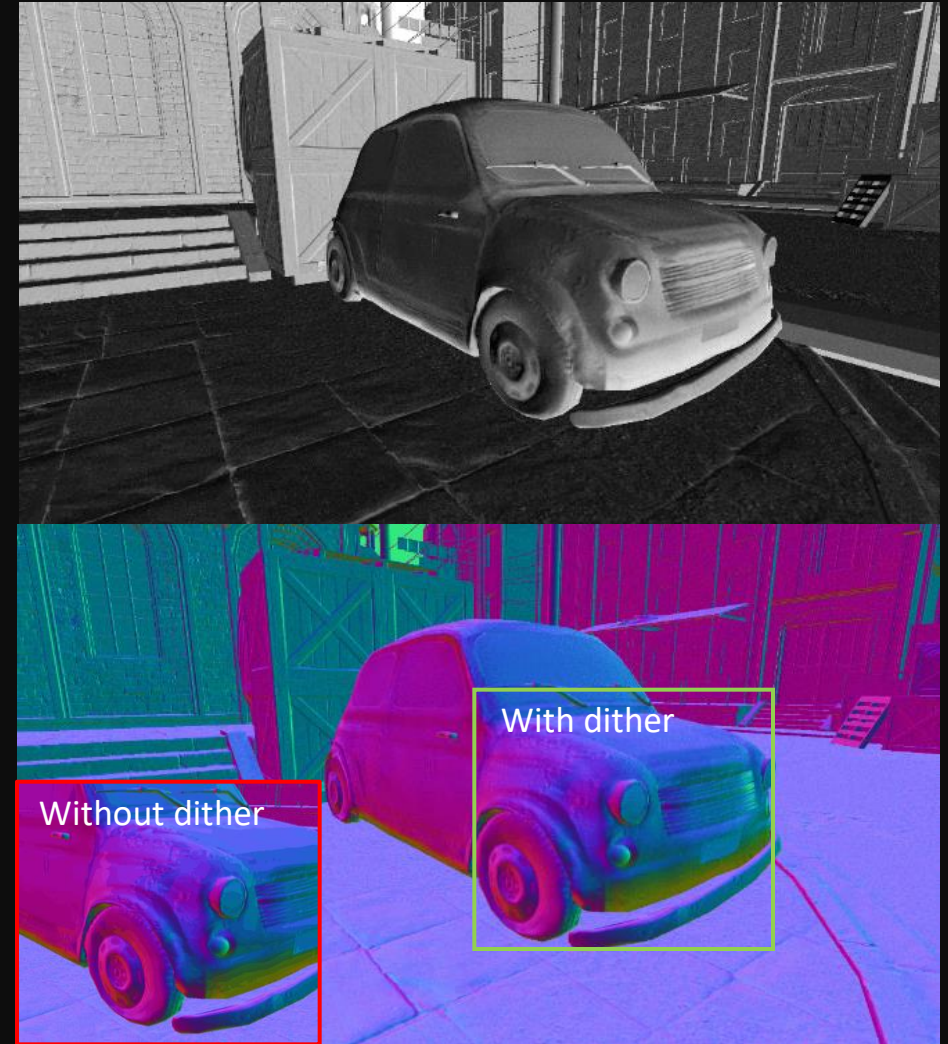
- **1 more texture:**

- Diffuse color: r8g8b8
- Normal: a8



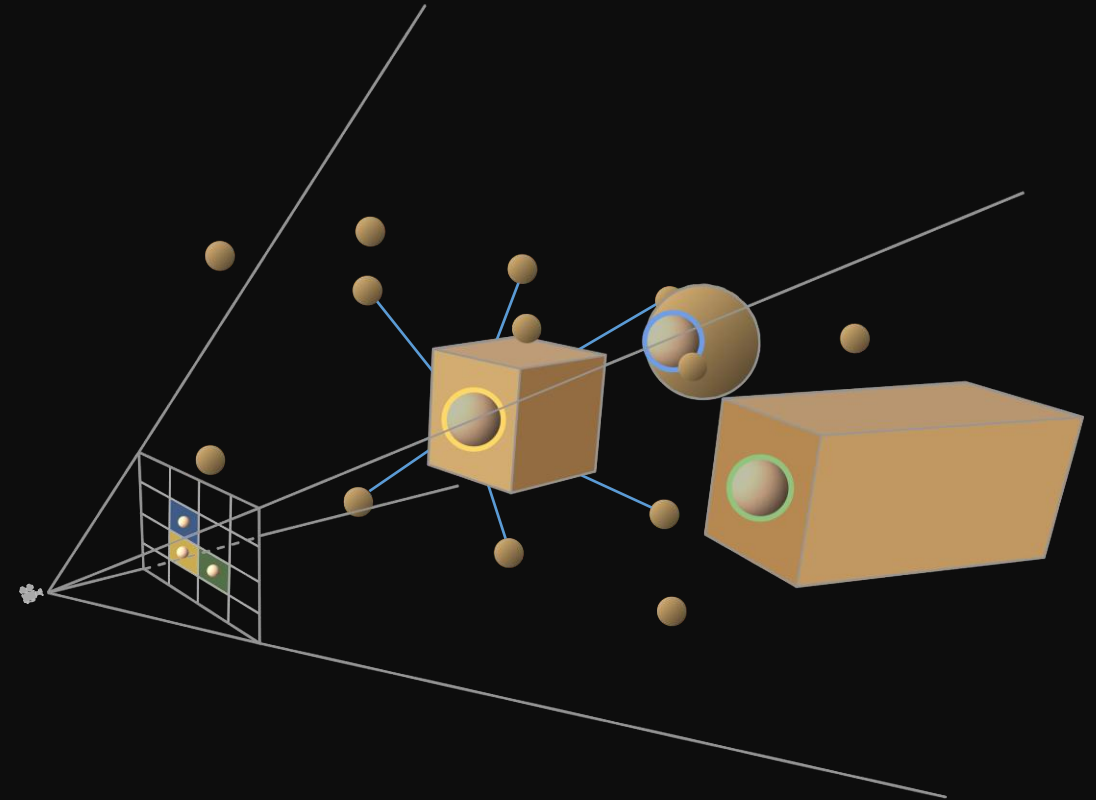
Deferred Screen Probe

- Lite Deferred pipeline
- 1 more texture:
 - Diffuse color: r8g8b8
 - Normal: a8
- **8 bits encoded normal:**
 - Transform world normal to view space
 - Add random noise to dither normal
 - Encode hemisphere normal to 8 bits



Deferred Screen Probe

- Format: **4x4-texels** hemisphere octahedral mapping
- 1. Split screen to 64x64-pixels tiles
- 2. Generate screen probes for each screen tile with jitter
 - Use Visibility Rays instead of VSM

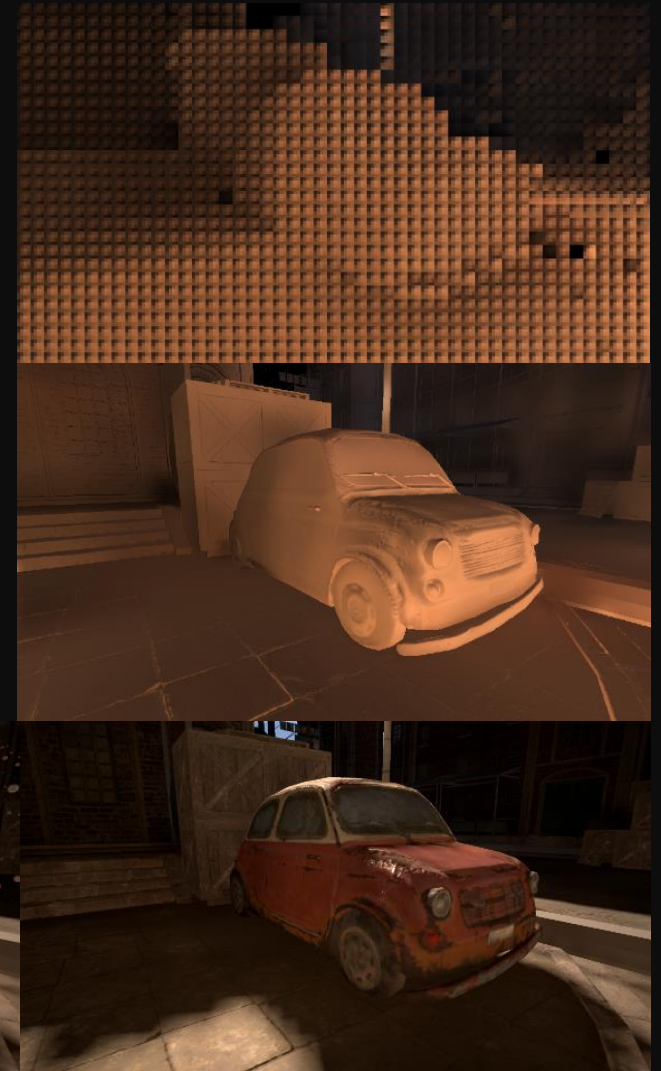


1 Screen Tile = 64x64像素 = 1 Screen Probe



Deferred Screen Probe

- Format: **4x4-texels** hemisphere octahedral mapping
- 1. Split screen to 64x64-pixels tiles
- 2. Generate screen probes for each screen tile with jitter
 - Use Visibility Rays instead of VSM
- 3. Gather screen probe in half resolution
 - Depth test and normal test
 - Temporal supersampling
- 4. Upsampe to full res
 - Bilateral filter



Final Gather

- **Half-Res screen irradiance:**

- Irradiance: R16G16B16
- Age: A16

- **Bilateral filter**

- gather screen probe
 - Depth test
 - Normal test

- **Temporal filter**

- Reprojection
 - Depth test
 - Convergence strategy with age

- **If all four screen probes were invalid:**

- Mark this tile
- Generate **1 more screen probe** for this tile next frame



Dynamic Objects

- No velocity map?
- Mask dynamic object:
 - Diffuse color: r8g8b7
 - Mask: m1
- For dynamic objects:
 - Gather 3D probe directly in half resolution
- For others:
 - Generate screen probes before drawing dynamic objects
 - If the history sample is marked dynamic: discard



Without velocity map



Add dynamic mask



Gen screen probe before dynamic



No artifact anymore!



Performance - Mobile

- **Hardware**

- Device: iPhone 13 pro
- Chip: A15

- **Base Pass**

- Before 2.20 ms
- After 1.88 ms
 - Skip indirect light calc.
 - Less ALU in basepass

Per Frame Pass	Cost (ms)
Generate Screen Probe	0.08
Gather Screen Probe	1.07
Indirect Lighting	1.00
Brick Collection	0.05
Injection	0.33
Irradiance Volume Updating	0.25
Hierarchical Structure Updating	0.40
Sum	3.18

Transient Pass	Cost (ms)
Hierarchical Structure Shifting	1.51
Irradiance Volume Shifting	0.21
Object Voxelization	0.08
Sum	1.79



Showcases









Feature Level: ES3_1
Level: HP_City_Main (Persistent)



Feature Level: ES3_1
Level: HP_City_Main (Persistent)





Feature Level: ES3_1
Level: HP_City_Main (Persistent)





Feature Level: ES3_1
Level: HP_City_Main (Persistent)



PC/Console Solution



Ray Tracing Again

- HDDA

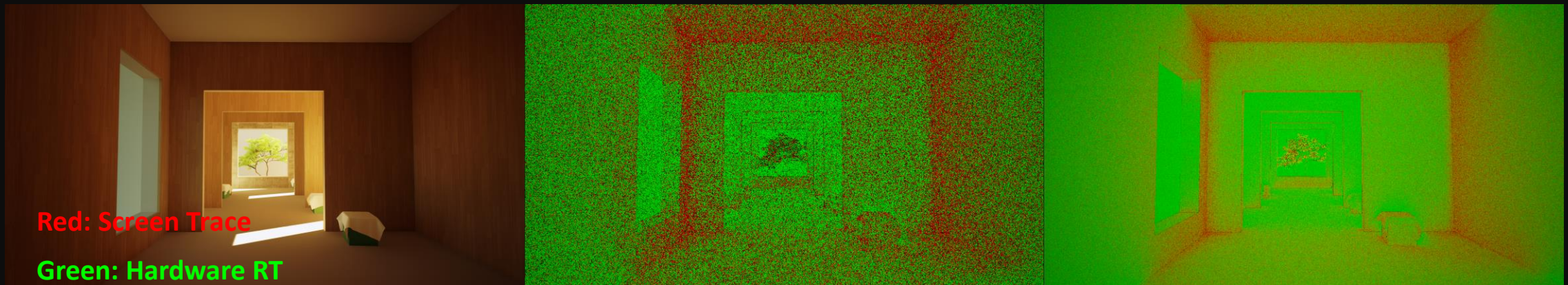
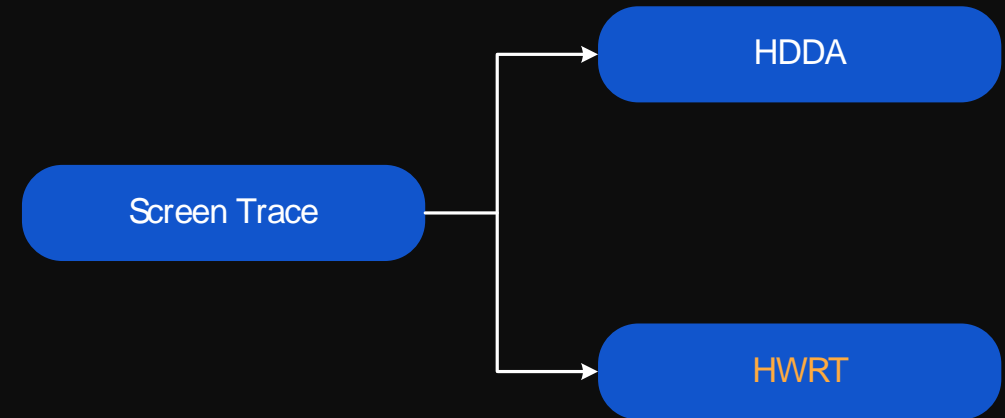
- Voxel Trace

- **Screen Trace** [Uludag 2014]

- HZB HiZ (diff resolution)
 - GBuffer friendly (geometry insensitive)

- **Hardware Ray Tracing**

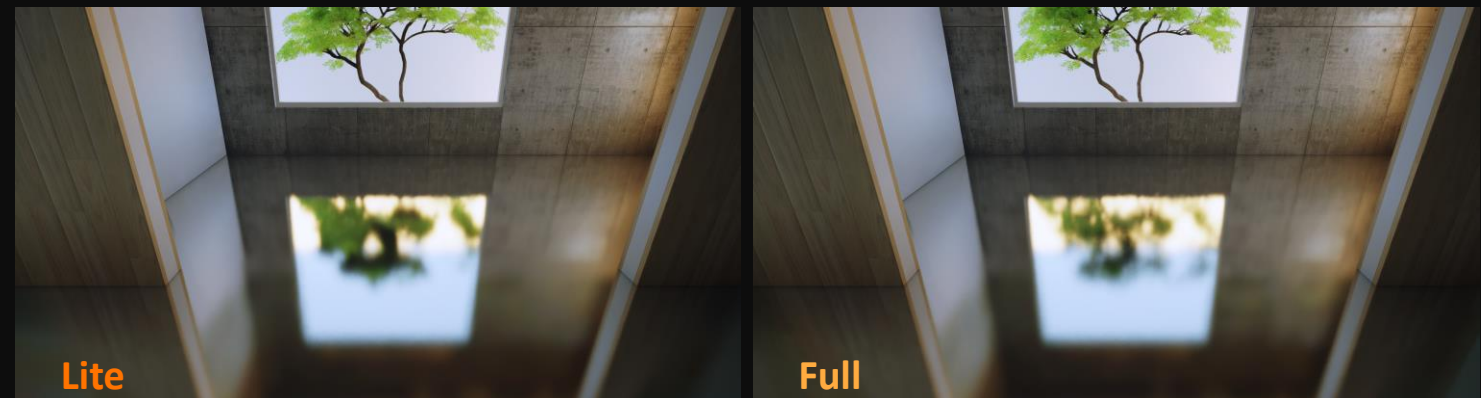
- PC/Console
 - Best quality



Ray Tracing – HWRT

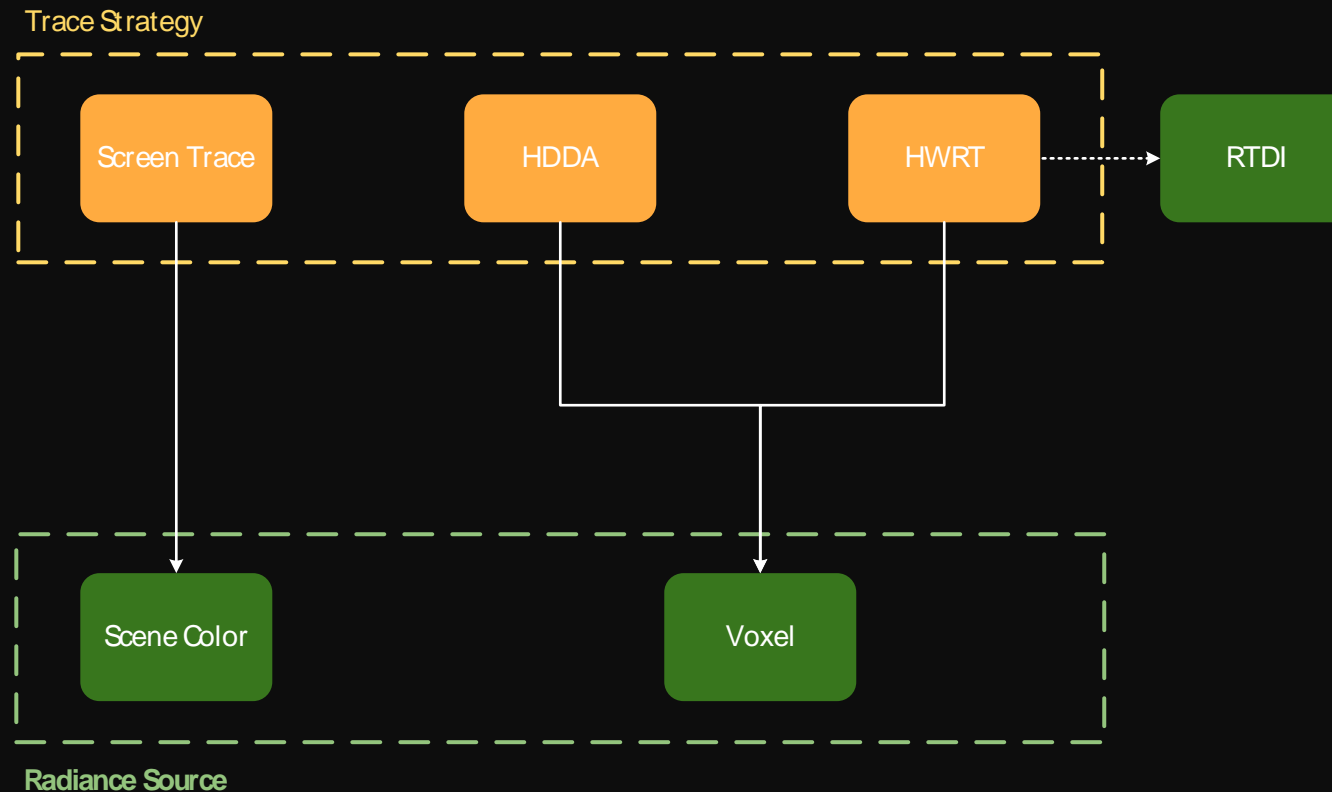
- **Lite/Full Pipeline**
- **Near/Far TLAS**
- **Self Intersection**
 - Short ray first
 - Cull back face
- **Translucent**
 - Directly penetrate
- **Mask**
 - Retrace (lite)
 - AHS (full)

Pipeline	Lite	Full
Material	Voxel	Dynamic Analytic
Indirect Radiance	Voxel	Voxel
Direct Radiance	Voxel	Voxel/RTDI
Any Hit Shader	No	Yes
Cost	Low	High
Application	Diffuse & Rough Specular	Glossy Specular



Ray Tracing - Summary

- Can use RTDI method for hit point direct lighting when HWRT is used
 - Accurate but expensive.
 - Indirect lighting always from the cache system.



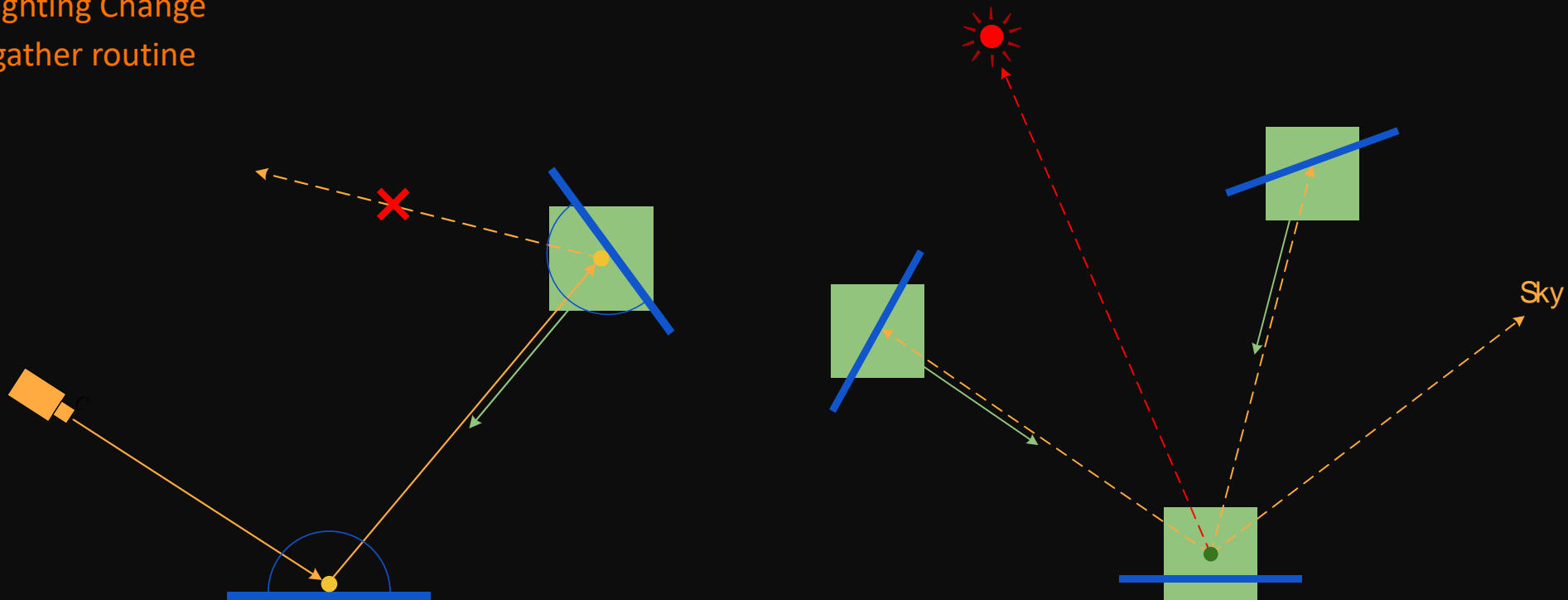
Voxel Cache for RT – Light Propagation

- Radiance Cache in Voxel

- Both direct and indirect
- Multi bounce support

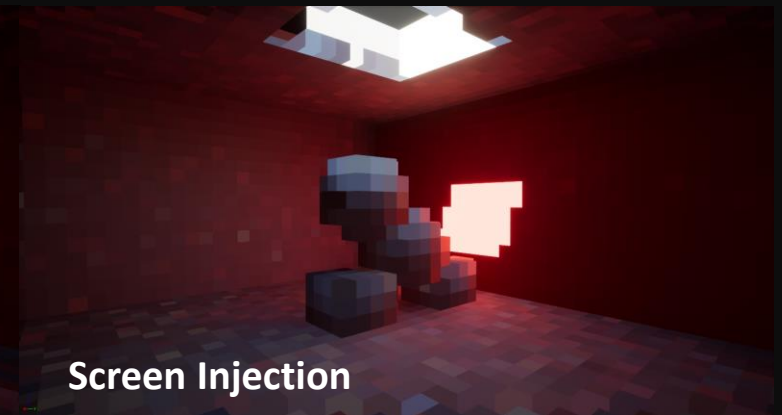
- How to Update

- Scene Lighting Change
- Simple gather routine



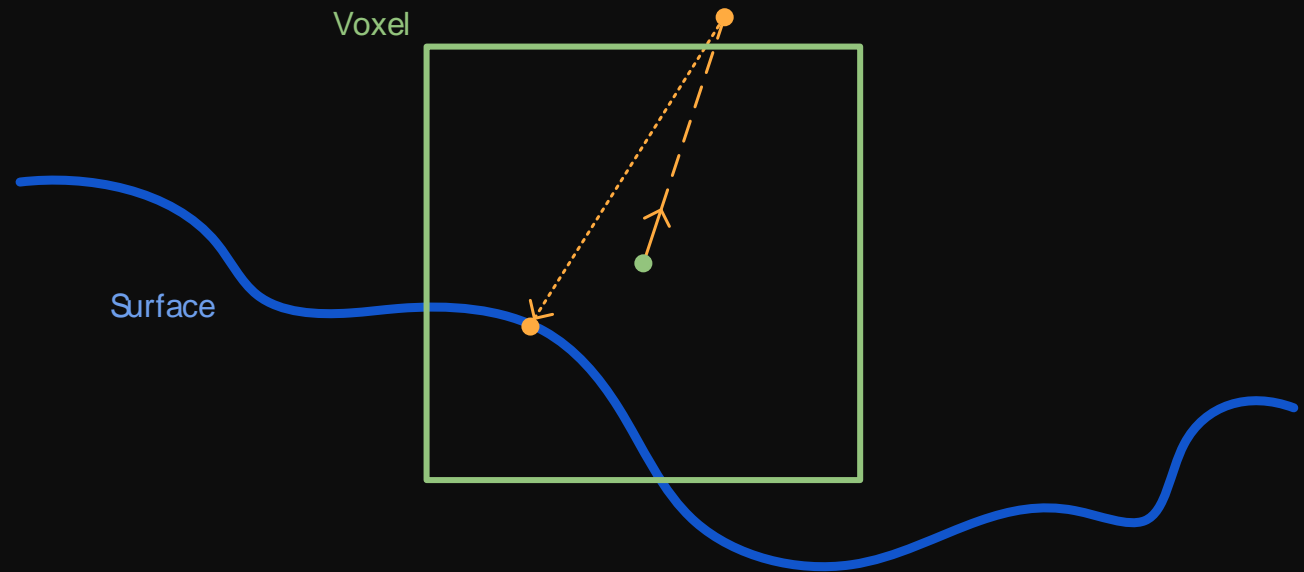
| Light Propagation

- **Direct Lighting**
 - RTDI (analytic, world light culling)
 - ReGIR solution in the future
- **Indirect Lighting**
 - Hemisphere gather with RIS
 - Temporal accumulation
- **Screen Injection**
 - High quality voxel lighting in view space



Light Propagation – Origin Geometry

- Where to trace from inside a voxel?
- 1. Offset along voxel normal
 - Voxel radius distance is enough
- 2. Trace short ray in voxel normal back hemisphere
 - Random direction in cone
 - Trace distance is voxel diameter
- 3. Judge whether in current voxel
 - Hit: fetch geometry from RTX
 - Miss: reconstruction from voxel

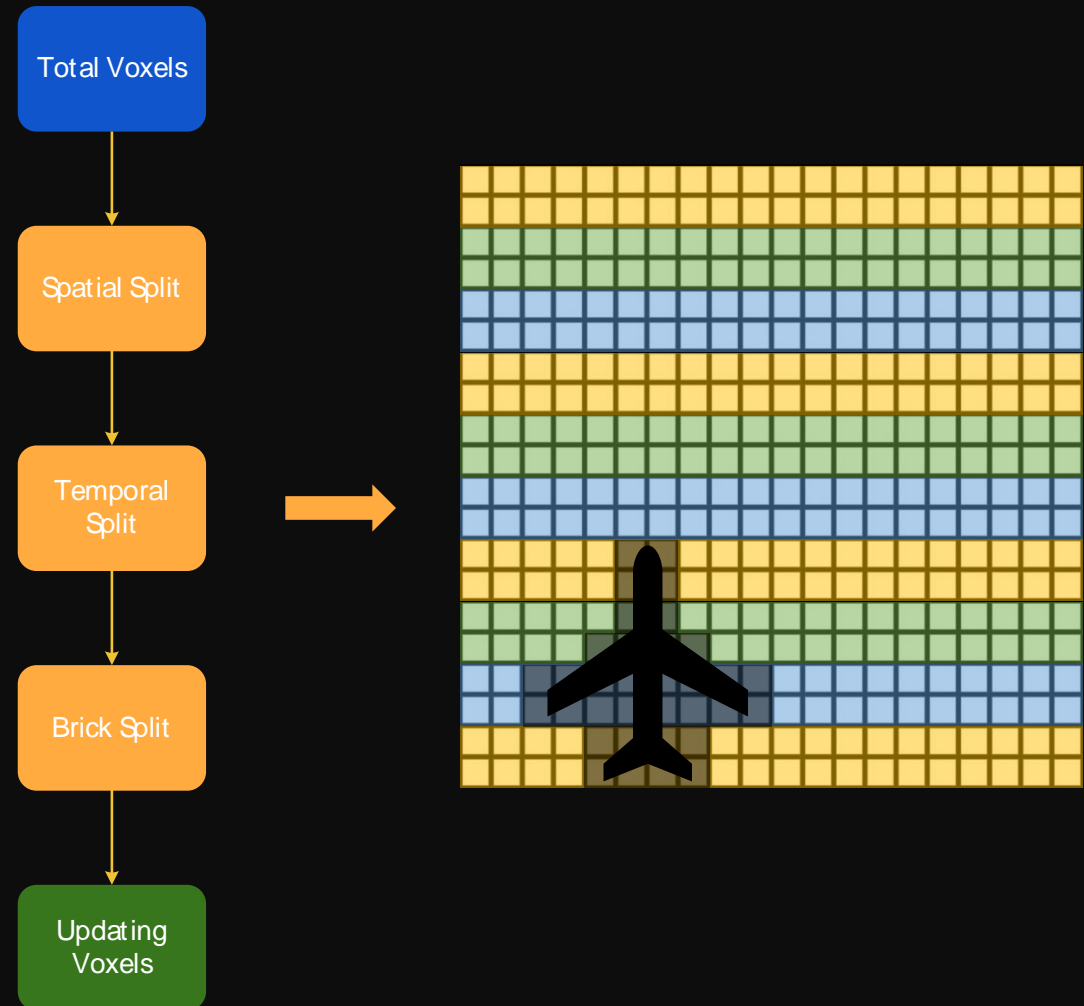
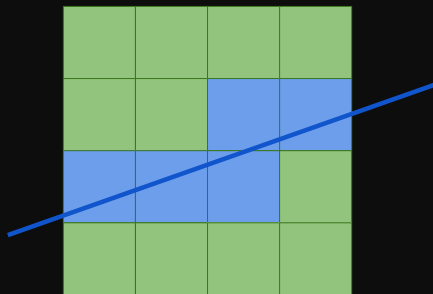


Light Propagation – Work Amortization

- Amortize computation cost

- Strategy

- Collect none empty bricks
 - Near field (9 frames, 300m x 300m x 150m)
 - Far field (121 frames)
 - Voxel mask (64 bits)
- Compute work
 - 16 threads for each brick
 - Uniform sample valid voxel
 - Insights: plane occupies 16 voxel in a brick at least
- Good cache performance, low divergence



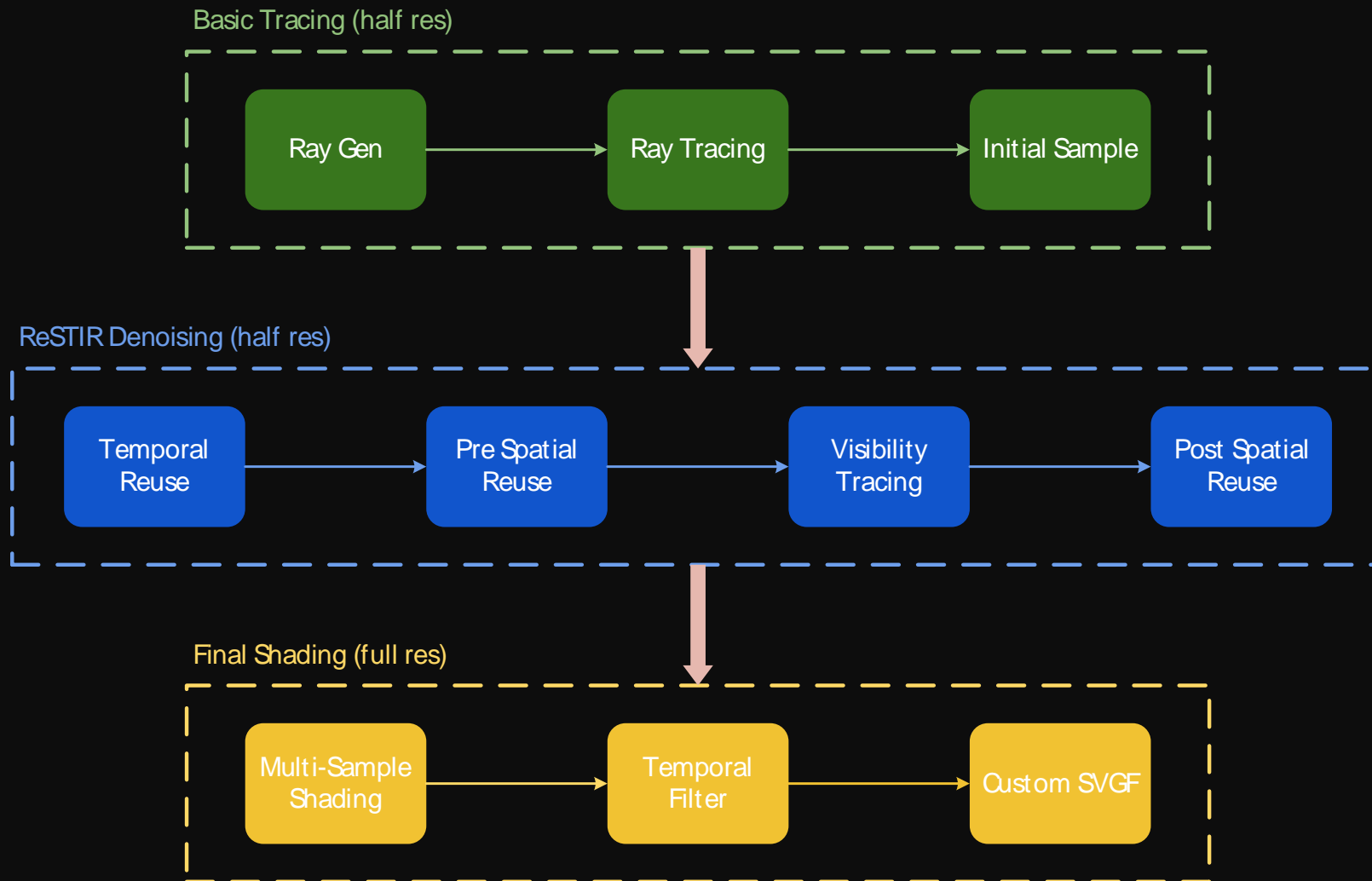
Scene Voxels Ready for GI!



Diffuse GI



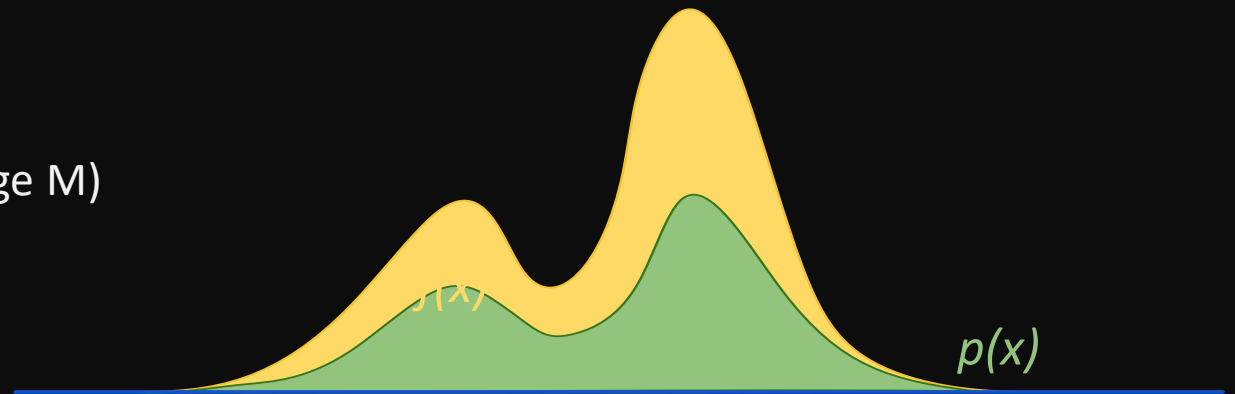
Pipeline



ReSTIR Basic

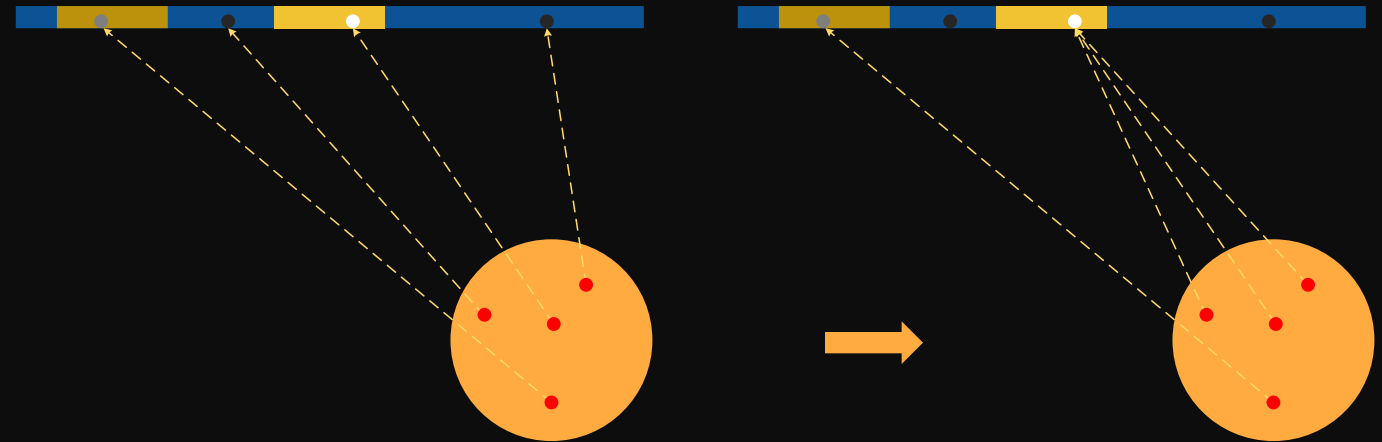
- [Bitterli 2020]
- **Resampled Importance Sampling**
 - Approximate better target PDF (M, N)
 - Source PDF: BRDF sampling
 - Target PDF: lighting function (without visibility)
- **Stream Reservoir**
 - Low memory footprint
 - GPU friendly
- **Abstract**
 - Keep most important sample
 - Support any BRDF
 - Large sample set from temporal & spatial (Large M)
 - Still need to reduce bias

$$\left\{ \begin{aligned} F &= \int_{\Omega} f(\omega) d\omega \approx \frac{1}{N} \sum_{i=1}^N \left[\frac{f(\omega_i)}{\hat{p}(\omega_i)} \frac{1}{M} \sum_{j=1}^M \frac{\hat{p}(\omega_{ij})}{q(\omega_{ij})} \right] \equiv \langle F \rangle_{ris} \\ W(\omega, z) &= \frac{1}{\hat{p}(\omega_z)} \left[\frac{1}{M} \sum_{i=1}^M w_i(\omega_i) \right] \\ \langle F \rangle_{ris} &= \frac{1}{N} \sum_{i=1}^N f(\omega_i) W(\omega, i) \end{aligned} \right.$$



ReSTIR GI

- [Ouyang 2021]
- **Temporal Reuse**
 - M clamping (20 frames)
 - Boiling filter
 - Suppress highlights and flickers
 - More stable
- **Spatial Reuse**
 - Group shared memory optimization
 - Introduce large bias!



Reservoir (16 bytes)	Surface (16 bytes)	Sample (32 bytes)
Bounce type flags	Surface type flags	Position
Selected sample index	Occluded sample index	Normal
M	Closest hit distance	Radiance
Weight	Linear scene depth	Hit distance
Target PDF	Normal	Custom data



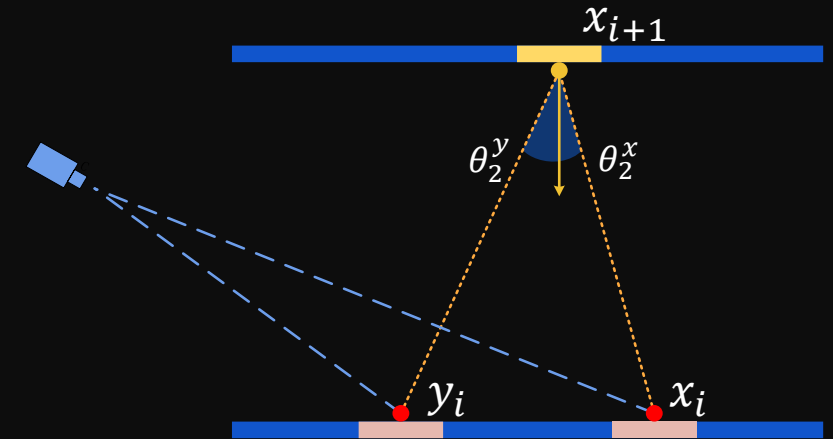
ReSTIR Intermediate Result

- Basic ReSTIR GI Processing
- $M = 1, N = 1$

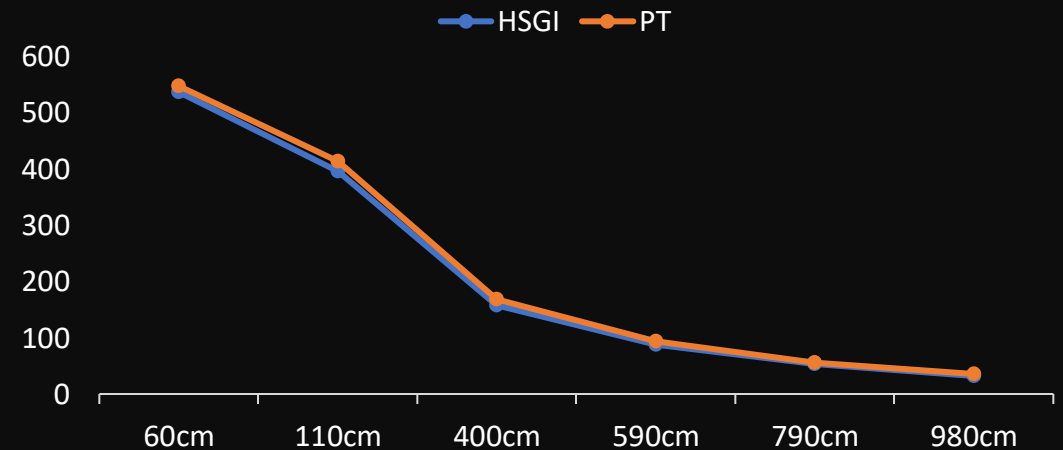


Jacobian

- Reconnection shift (solid angle PDF conversion)
 - [Ouyang 2021], [Lin 2022]
- Critical for GI detail (natural AO effect)
- Diffuse energy bias less than 15% comparing PT
- Suffer from singularity, **bound max value (not min)**
 - Prefer darken bias than lighten bias



Energy Decay With Distance



Bias Analysis

- Too Much Bias!
- Lack Indirect Detail!
- Poor Quality!



Naïve ReSTIR GI



PT



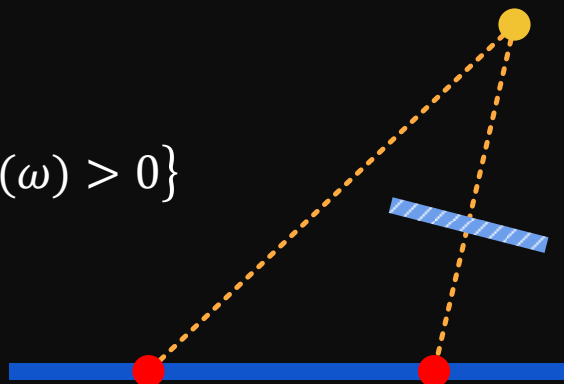
UE5



Bias Analysis

- **Target PDF without visibility**
 - Lost indirect shadow, contact AO, lighting detail
- **Dominated bias**
 - Multi-pass spatial reuse
- **Heuristic neighbor selection**
 - Already done, not enough
- **Trace rays for each step**
 - Too costly

$$\left\{ \begin{array}{l} \mathbb{E}[W(\omega, z)] = \frac{1}{p(\omega_z)} \frac{|Z(\omega_z)|}{M} \\ Z(\omega) = \{j \mid 1 \leq j \leq M \text{ and } q_j(\omega) > 0\} \end{array} \right.$$



- **Biased ReSTIR Version**
 - More spatial reuse, more stable, more bias
 - Lower quality comparing to UE [Wright 2022]
 - Screen probe + bent normal
- **Unbiased ReSTIR Version** [Bitterli 2020]
 - Too noise under high frequency lighting area
 - No enough successfully reused sample
 - If you do nothing, then it is unbiased!
- **What we want?**
 - Keep high frequency indirect shadow detail
 - Real time and stable
 - Not simple screen AO solution



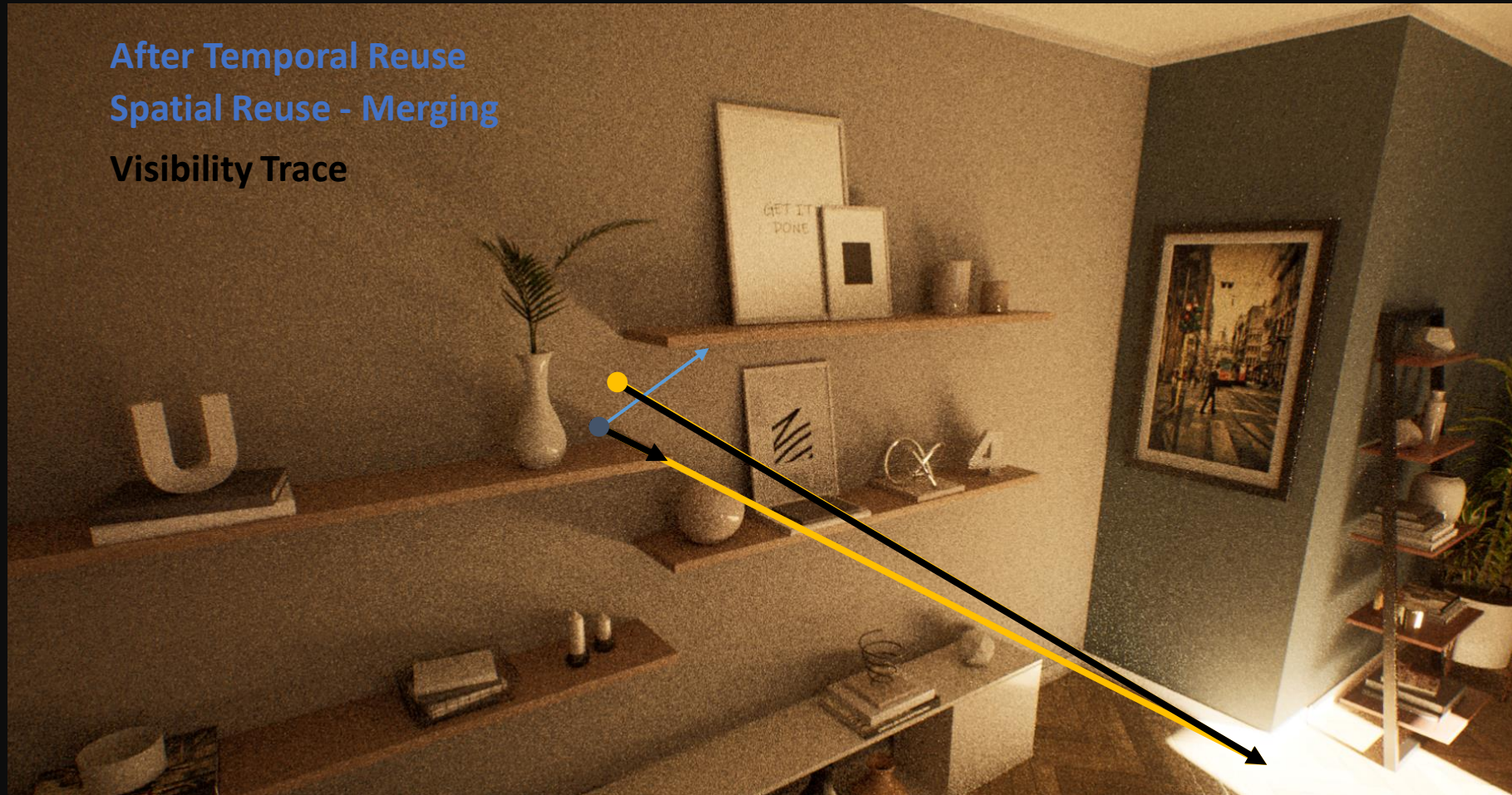
Question?

- **Visibility bias.** Bright samples lighten shadowed area. How to extract indirect shadow information?



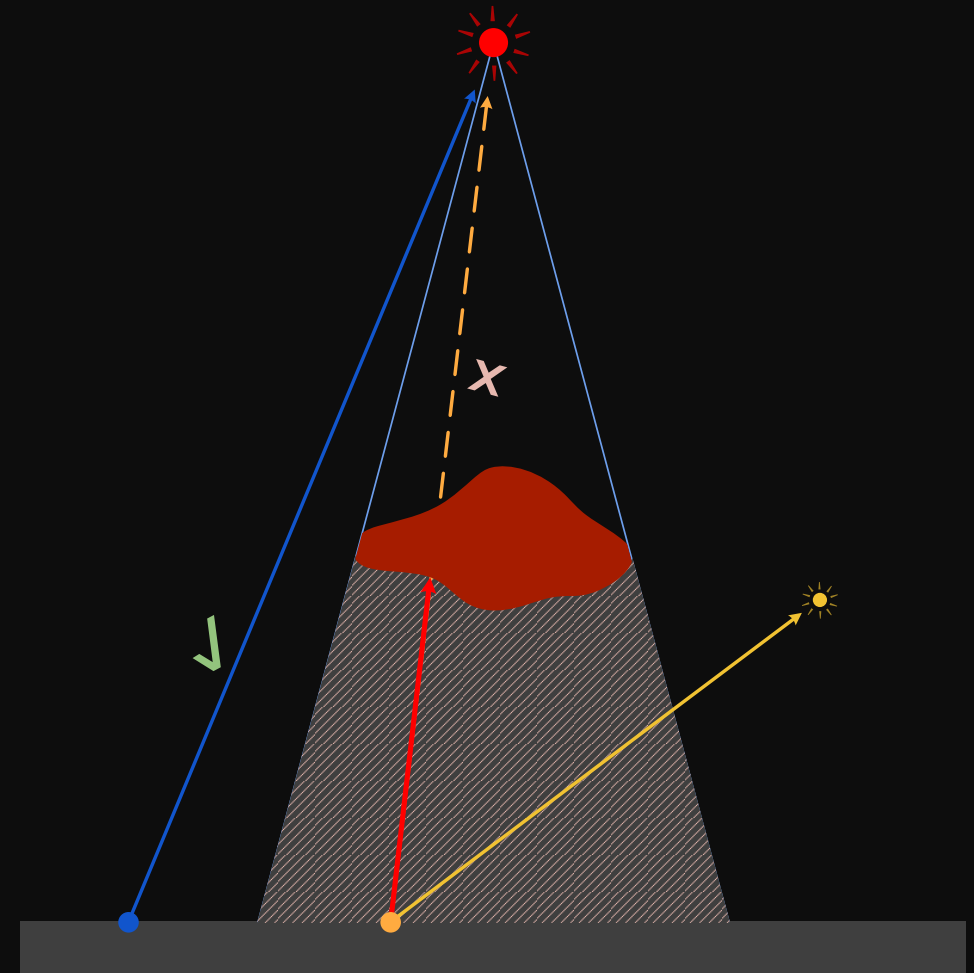
Idea!

- **Simple 1 visibility trace may help!** Restore to low bias temporal reuse result. Pretty efficient!



ReSTIR GI Shadow

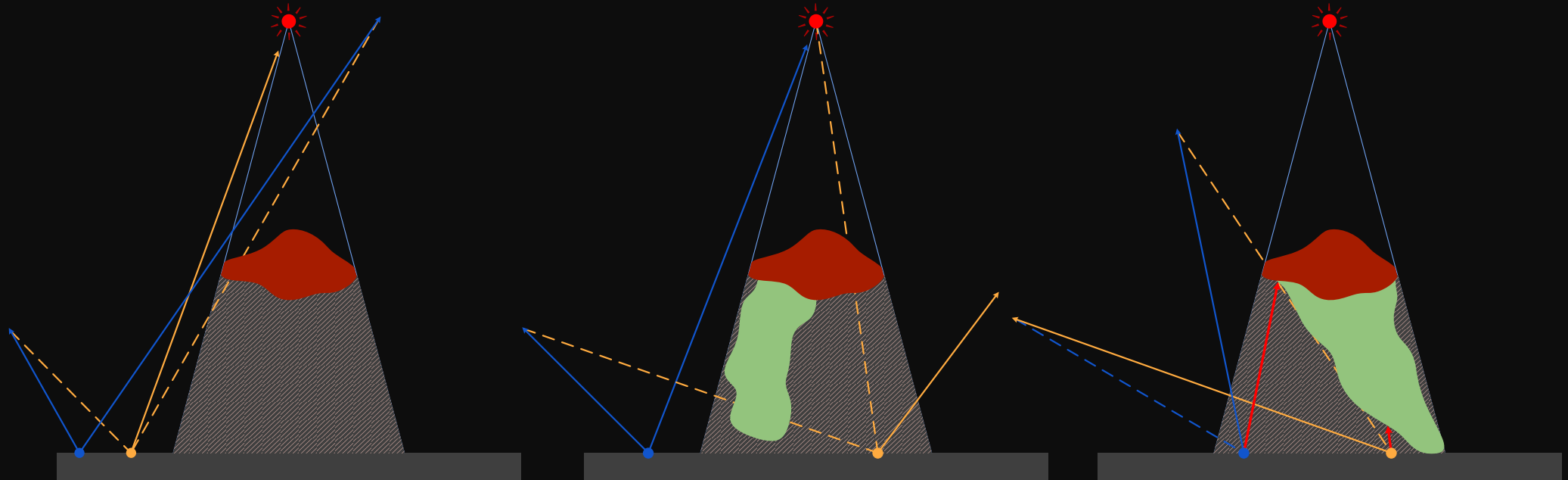
- 1. Naïve ReSTIR select most important sample
 - The brightest local virtual light (sample)
- 2. Trace **1 ray** after the last spatial reuse
 - From pixel position to sample position
 - Indirect light shadow info
- 3. Check visibility
 - If miss:
 - Valid sample
 - If hit:
 - Store occlude distance and direction
 - Restore to temporal reuse result (less bias)
- 4. Post spatial reuse
 - Heuristic neighbor selection with shadow condition
 - Whether occluded (whether in shadow)
 - Occlusion direction and distance



ReSTIR GI Shadow – Post Spatial Reuse

- Heuristic neighbor selection with shadow condition

- 1. Both outside shadow (low risk, accept)
- 2. One inside shadow, one outside shadow (high risk, reject)
- 3. Both inside shadow (judge occlusion direction and distance)
 - *if $\text{dot}(\text{CenterOccDir}, \text{SampleDir}) > a \parallel \text{NeighborOccDist} > \text{CenterOccDist}$*
 - *return $\text{CenterOccDist} > b \cdot \text{SampleDist}$ (high risk)*
 - *else return true (low risk)*



PT



UE5

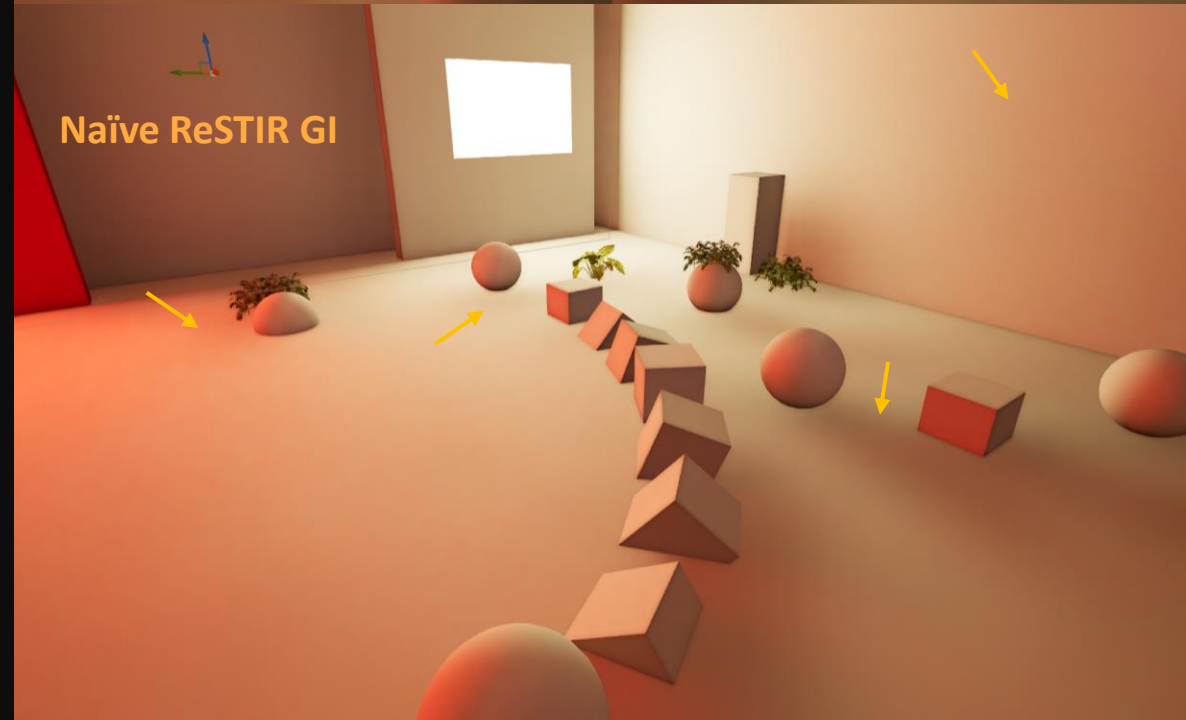


ReSTIR GI Shadowing



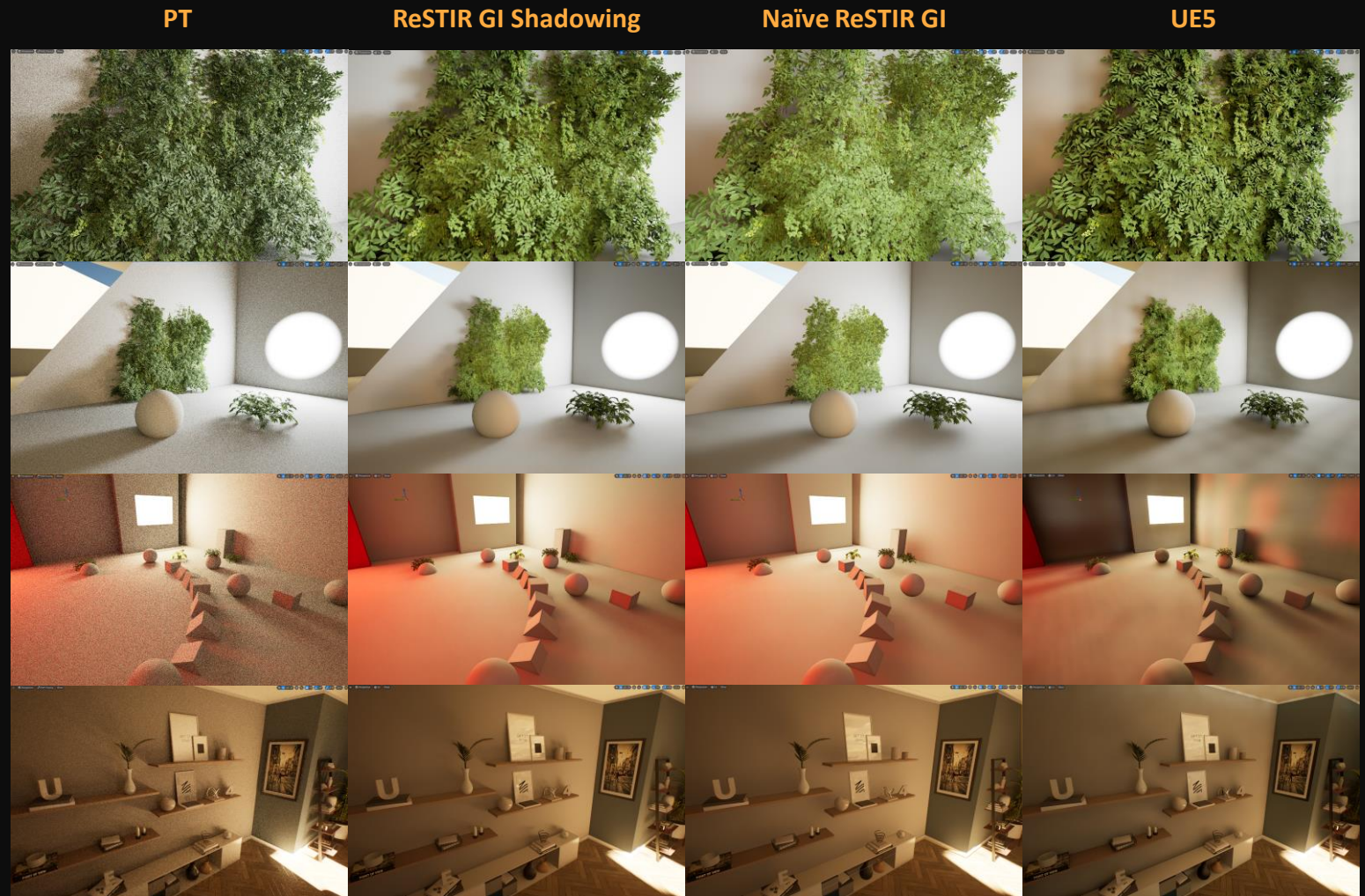
Naïve ReSTIR GI





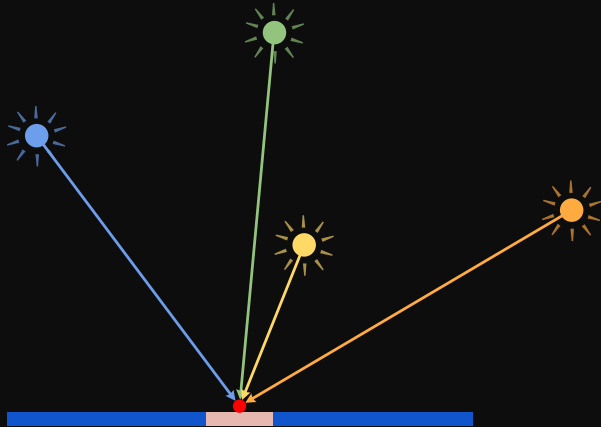
ReSTIR GI Shadow Summary

- Keep indirect shadow
- High frequency detail
- Indirect shadow extraction
- 1 ray per pixel
- Enough for real time



Color Noise

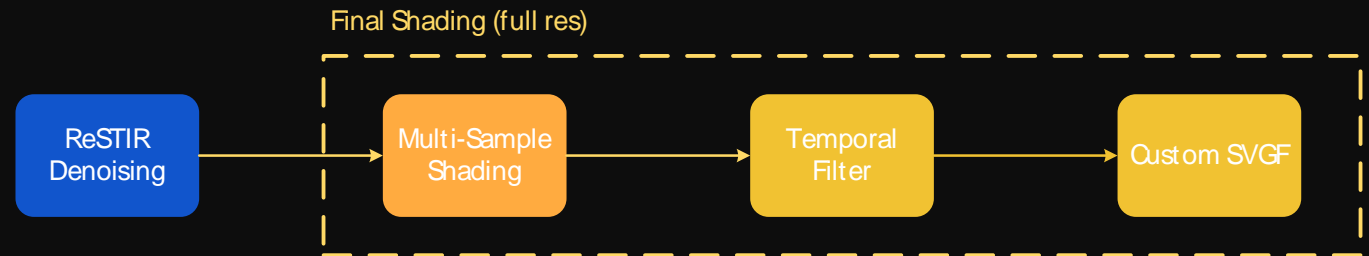
- Current ReSTIR pipeline: large M small N
- 1 sample is not enough for final shading
- Target PDF is scalar without spectrum info
 - Save ALU and Memory
 - If RGB, **all the things x3**
- Should increase N!
 - Temporal filter is slow for N
 - New area from occlusion always own small N



$$F = \int f(x)dx \approx \frac{1}{N} \sum \left[\frac{f(x_i)}{\hat{p}(x_i)} \frac{1}{M} \sum \frac{\hat{p}(x_j)}{p(x_j)} \right]$$

Multi-Sample Shading

- Just like reservoir spatial reuse
- Theory support
 - Use neighbor reservoir to merge empty reservoir
- Shade current pixel with neighbor sampling info
 - 3x3, 5x5, 7x7
- Increase N efficiently
- Suppress color noise
- Support upsampling natively
 - Keep full screen detail
 - material and geometry



Final Processing

- Temporal Supersampling

- Reprojection
- Depth and normal occlusion rejection
- Frame number accumulation
- Color clamping [NRD]
 - Long history (50 frames)
 - Short history (10 frames)
 - Short history 5x5 neighbor info
 - Variance clamping

- Custom SVGF [Schied 2017]

- Kernel weight
- Normal weight
- Depth weight
- Luminance weight



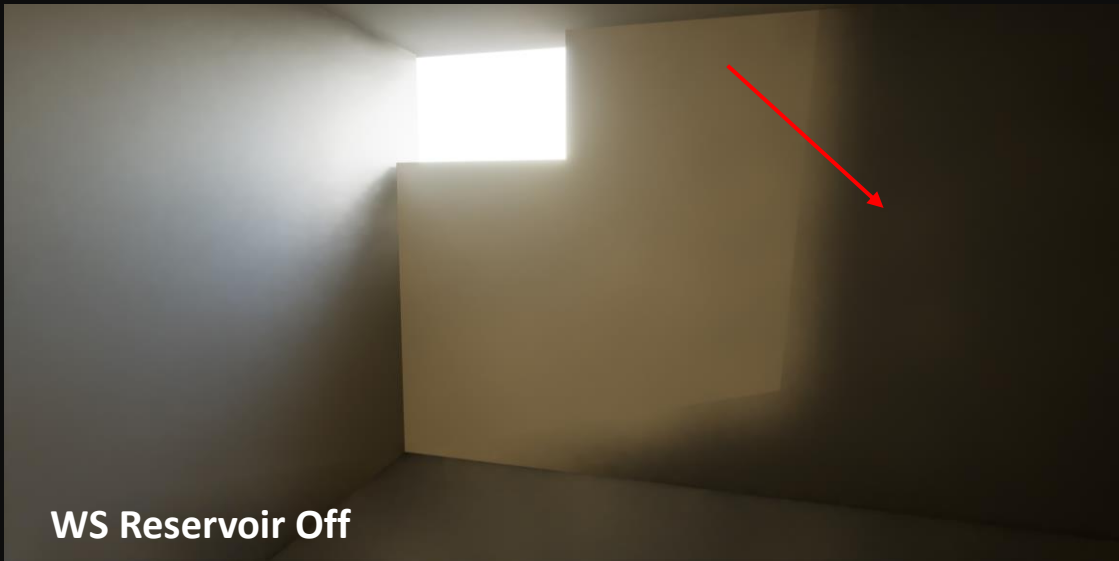
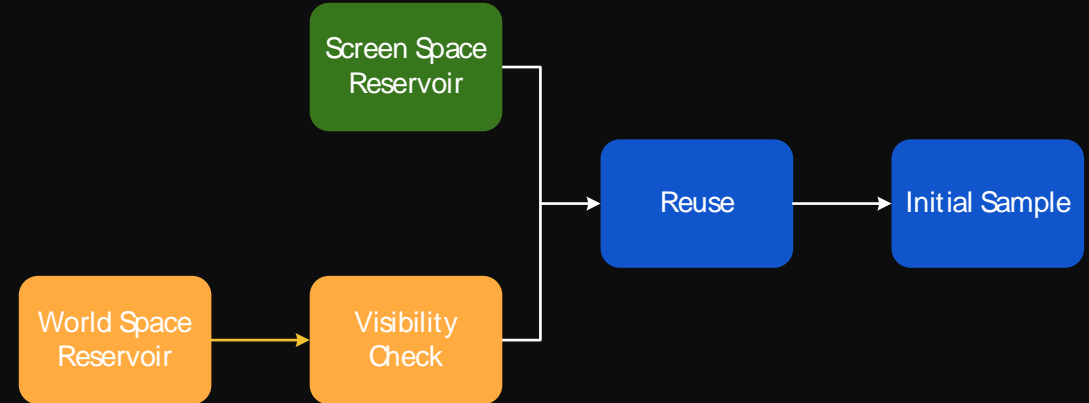
World Space Reservoir

- **Screen Space ReSTIR Problem**

- Disoccluded areas & fast camera movement
- No enough history samples
- Different degrees of convergence

- **World Space Reservoir (ReGIR)** [Boksansky]

- Generated by light propagation, stored in brick level
- Trace 1 ray to reduce visibility bias
- Fast converge, more stable



Trace Probe Mode

- **Critical Scene Lighting**

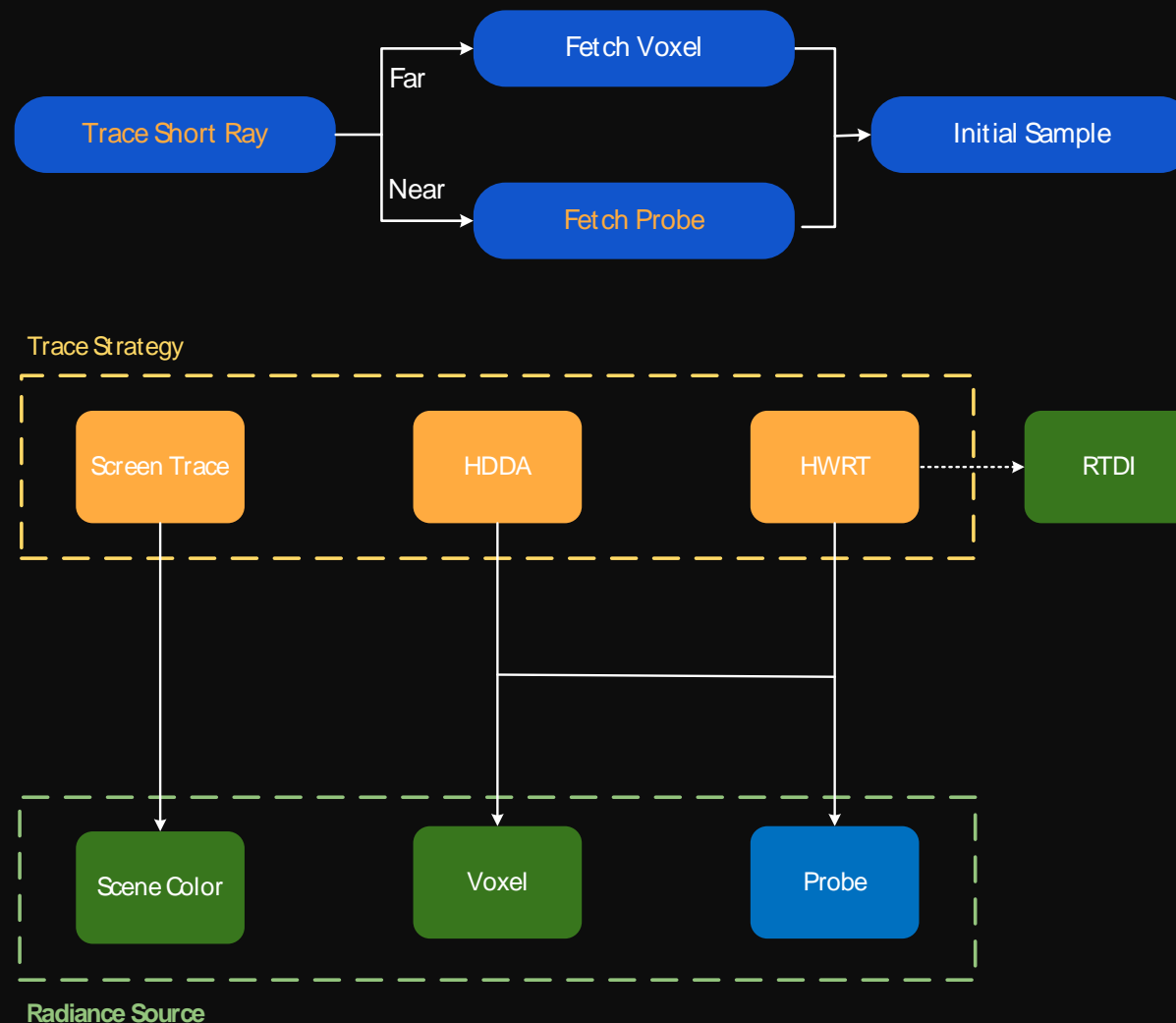
- Main bounce number ≥ 4
- Bright but small light source

- **Trace Probe**

- Short ray (= probe size)
- Fetch radiance from probe
- Smooth and stable
- Apply to far distance position

- **Probe Format (PC/Console)**

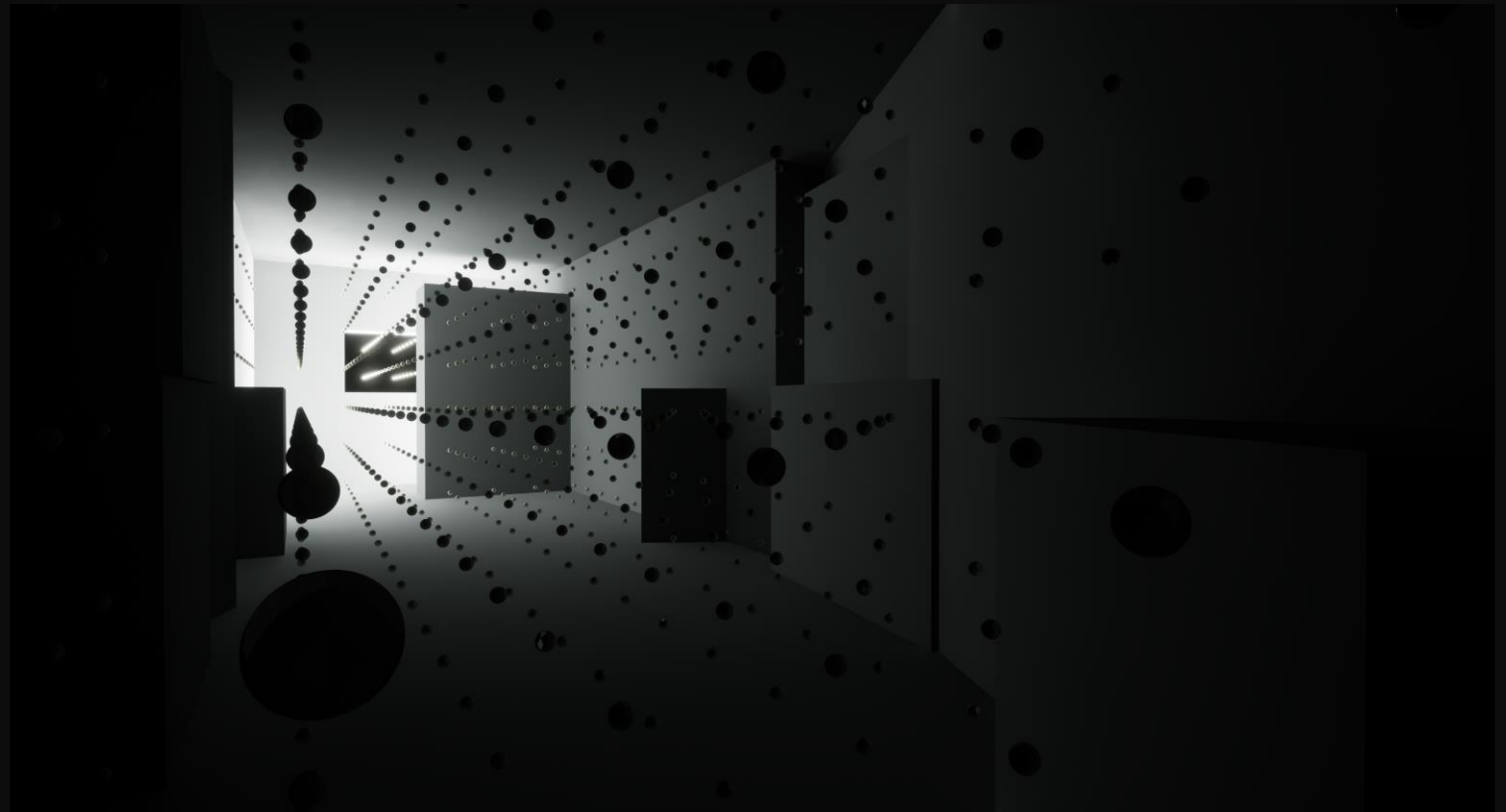
- **Radiance:** 6x6-texels octahedral mapping
- Visibility: VSM distance
- Relocation Info: index and alpha
- State and age



Trace Probe Mode

- **Best Practice**

- Trace ray mode as default for high quality
- Switch trace probe in critical scene for stability
- Dynamic switch for local areas



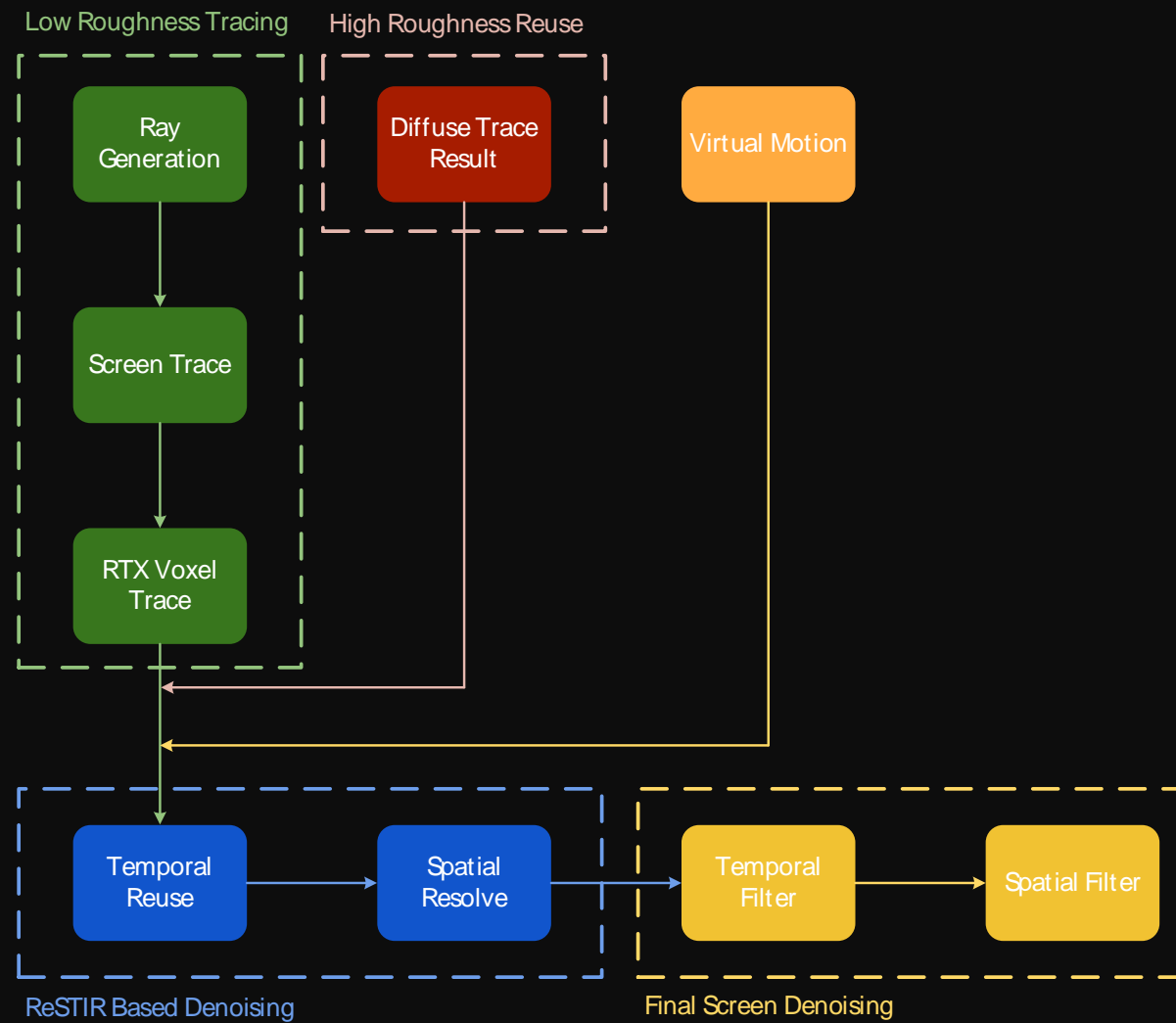
Specular Reflection



Reflection Is Important for GI



Pipeline



Ray Generation

- **Full range roughness support** [Kajiya]
 - Specular Ray
 - Roughness below 0.4 (GGX VNDF)
 - Screen trace (full res HZB)
 - HWRT (Lite/Full)
 - Diffuse Ray
 - Roughness above 0.4
 - Reuse diffuse tracing result
 - Performance opt
 - **Uniform denoising**
- **Canonical Sample Coverage** [Lin 2022]
 - 2x2 jitter
 - Trace specular ray for 1 pixel
- **Lerp to Diffuse GI**
 - For high roughness above 0.8



ReSTIR Denoising - Temporal

- Combine traced specular and rough specular (uniform denoising routine)
- Virtual motion support [NRD]
- Boiling filter (suppress specular firefly but lost energy)
- **ReSTIR PDF stuff adjustment (match GGX)**
 - High roughness: reconnection shift; low roughness: half vector copy shift (not trace ray)

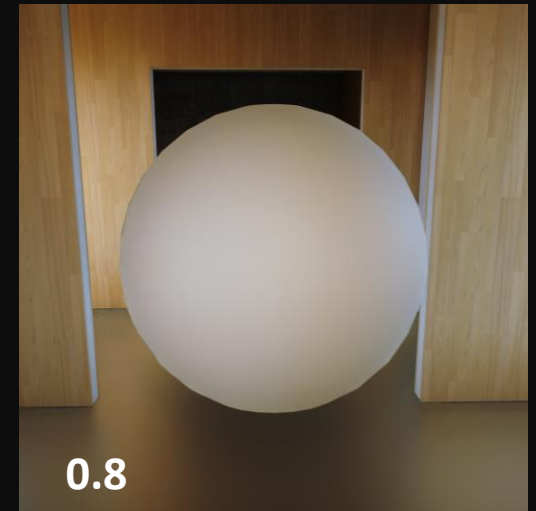
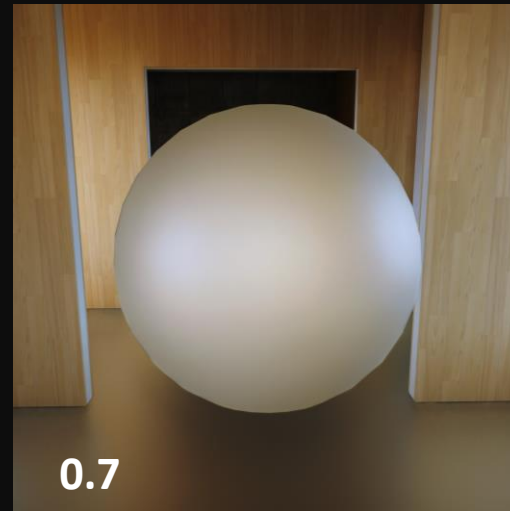
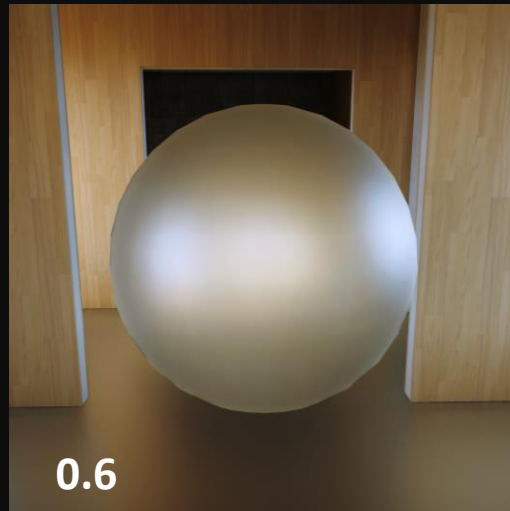
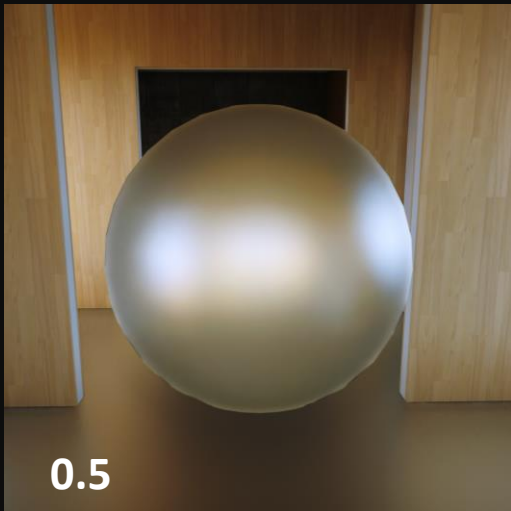
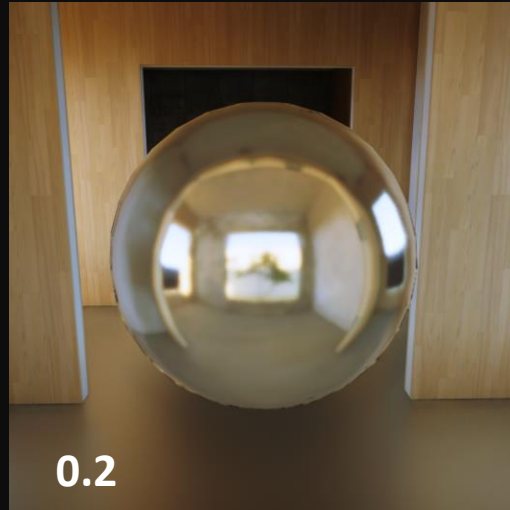


ReSTIR Denoising – Spatial

- **World space spatial kernel (ReBLUR)** [Zhdan 2021]
- Resolution upscaling
- **ReSTIR Multiple shading shading**
 - Traditional IS resolve process not work (only BRDF sampling, too noisy) [Stachowiak 2015 & 2018]

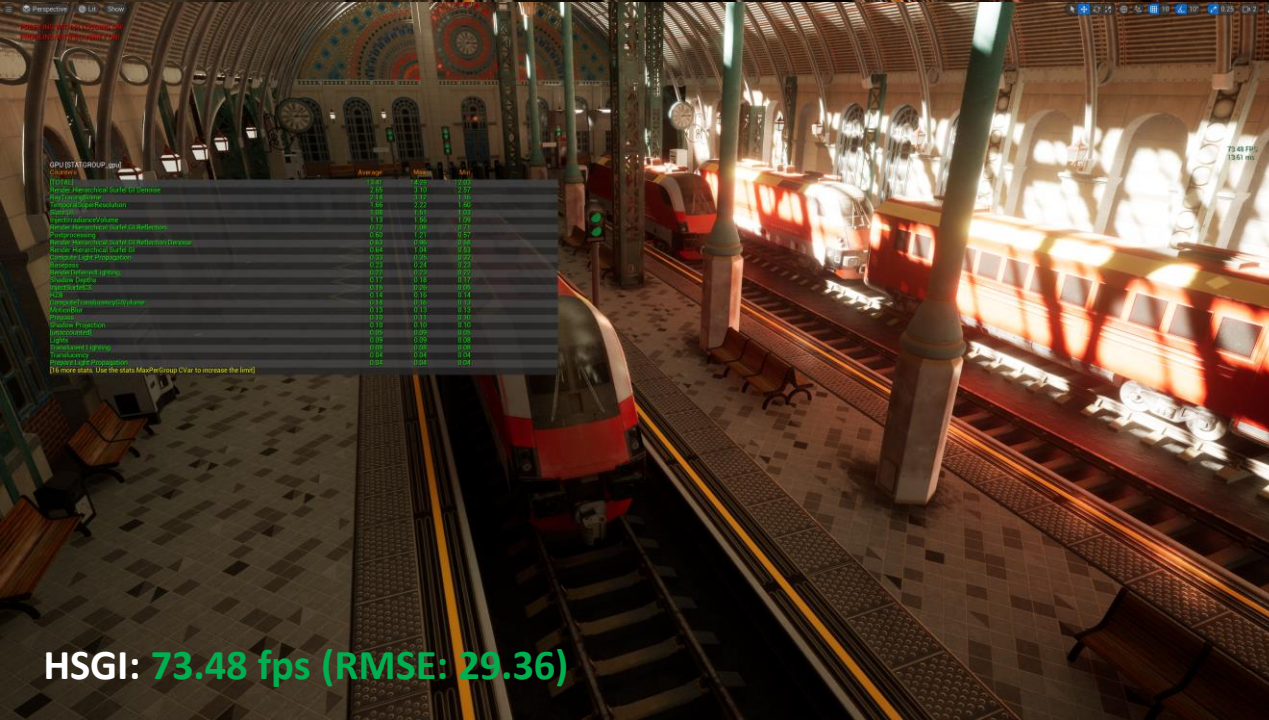


Roughness response



Performance





Performance - PC

• Hardware

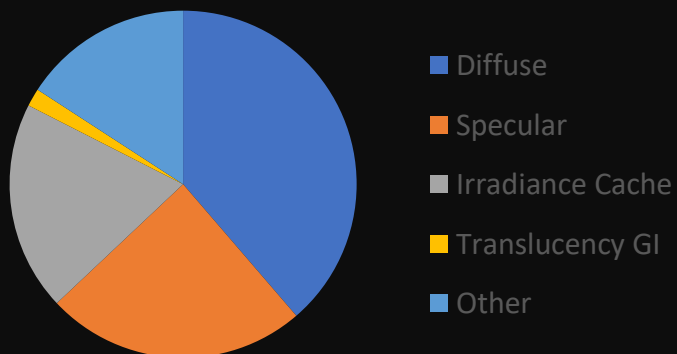
- CPU: AMD Ryzen Threadripper 3970X
- GPU: NVIDIA GeForce RTX 3080

• Resolution

- TSR: 3840 x 2160
- Render: 1932 x 1084
- Tracing: 966 x 542

• FPS

- None: 118.51
- HSGI: 73.48
- UE: 59.62



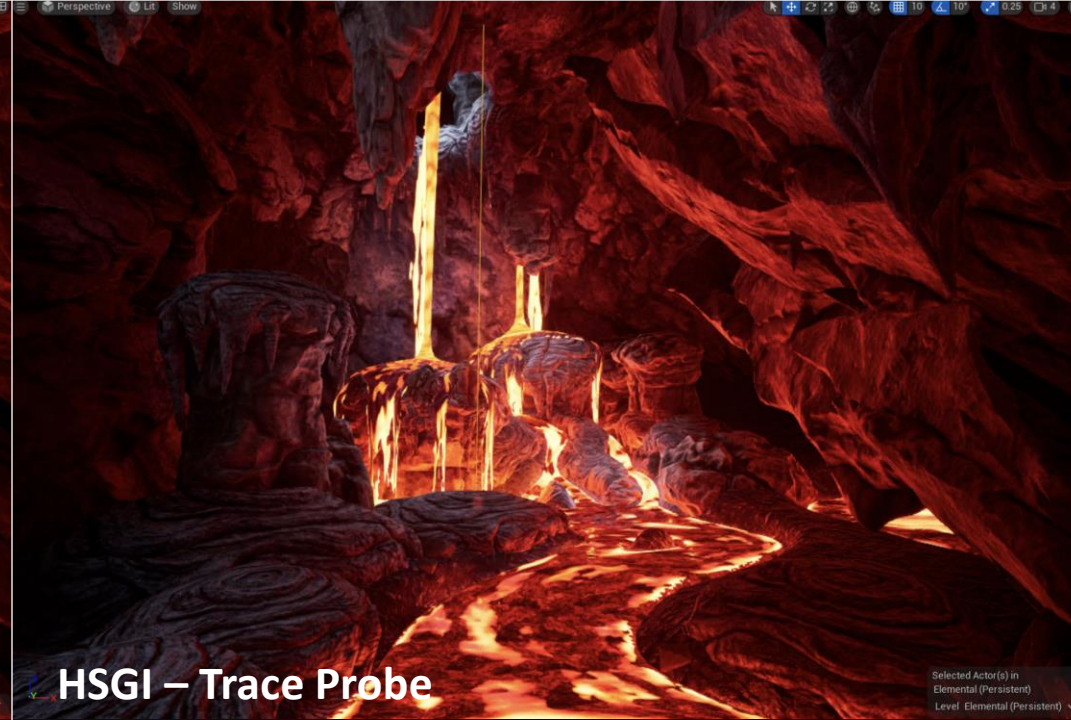
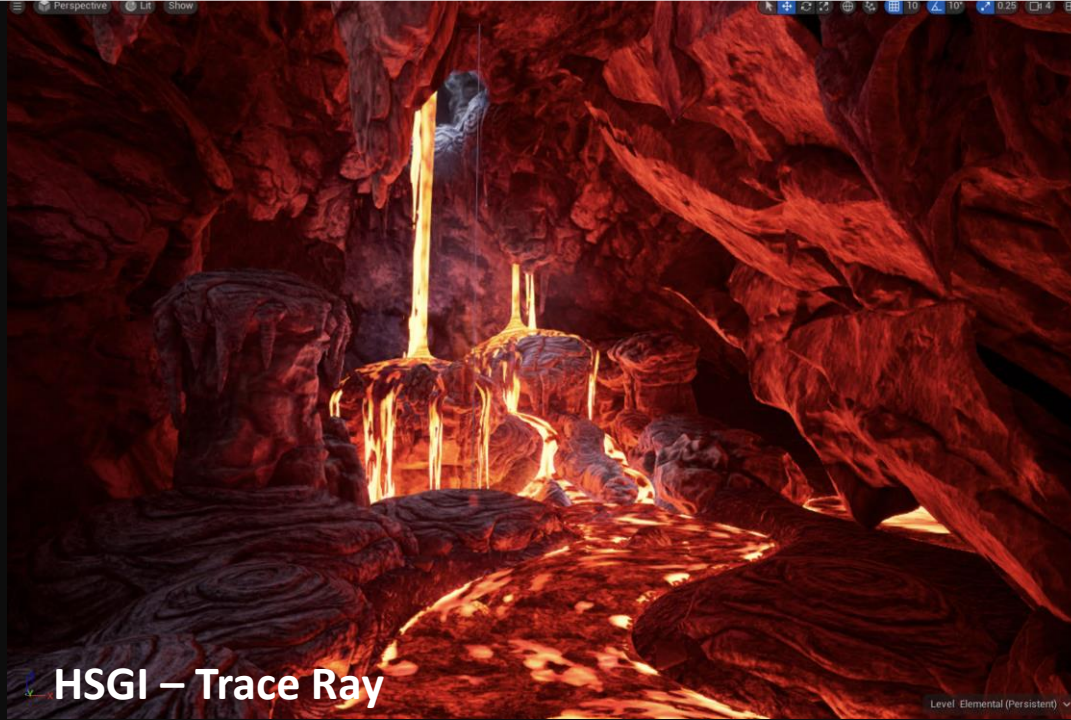
UE5 Pass	Cost (ms)
Scene Update	0.06
Scene Lighting	1.14
Screen Probe Gather	1.83
Reflections	1.70
Ray Tracing Scene	4.90
Other	7.14 (+9.63)
Sum	16.77

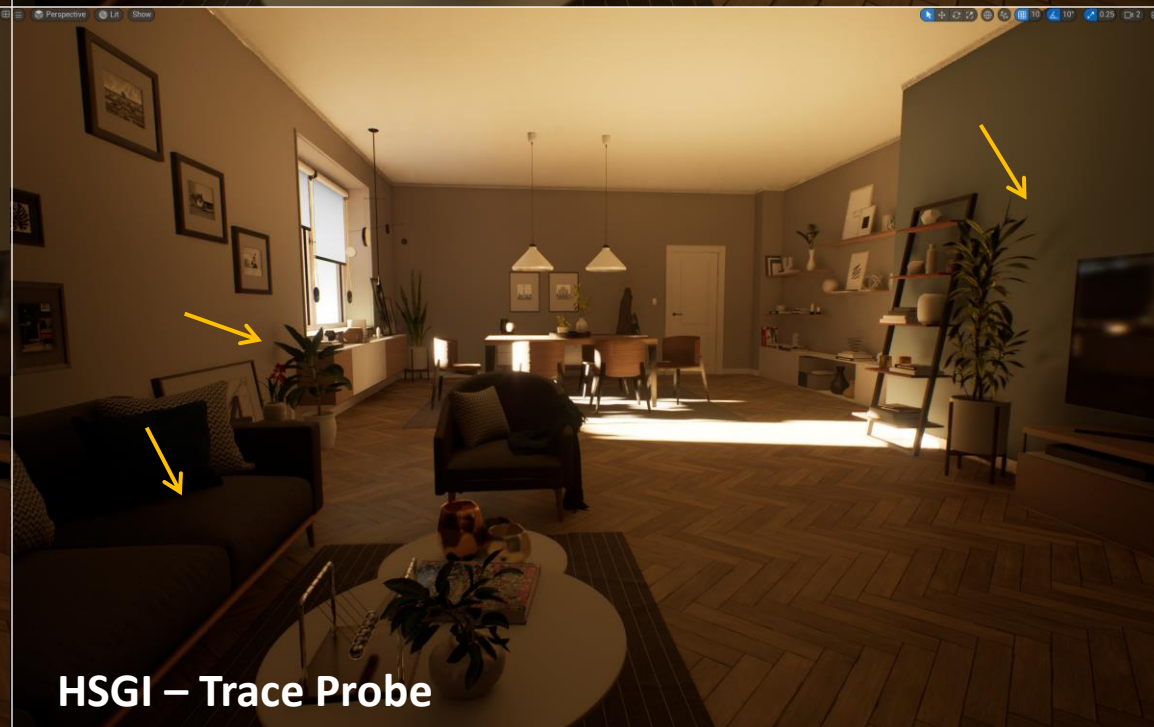
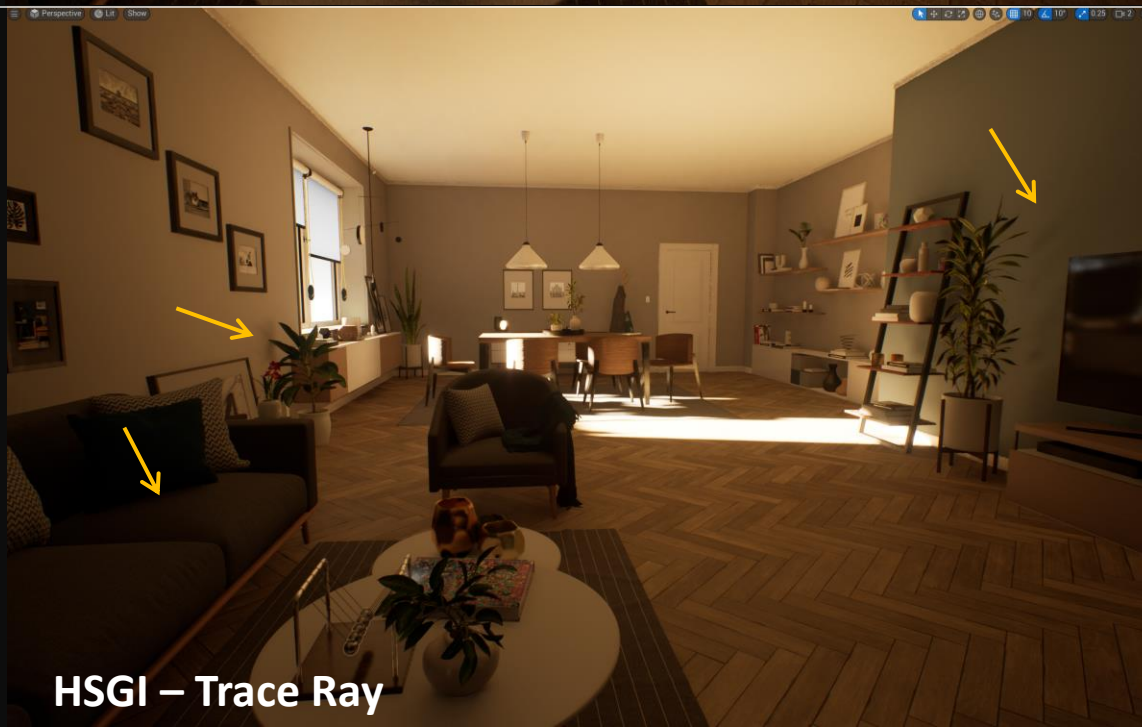
HSGI Pass	Cost (ms)
Diffuse Trace	0.64
Diffuse Denoise	2.65
Specular Trace	0.72
Specular Denoise	0.63
Light Propagation	0.33
Inject Irradiance Volume	1.13
Translucency GI Volume	0.14
Screen Inject Surfel	0.15
Ray Tracing Scene	2.14
Other	5.08 (+8.53)
Sum	13.61

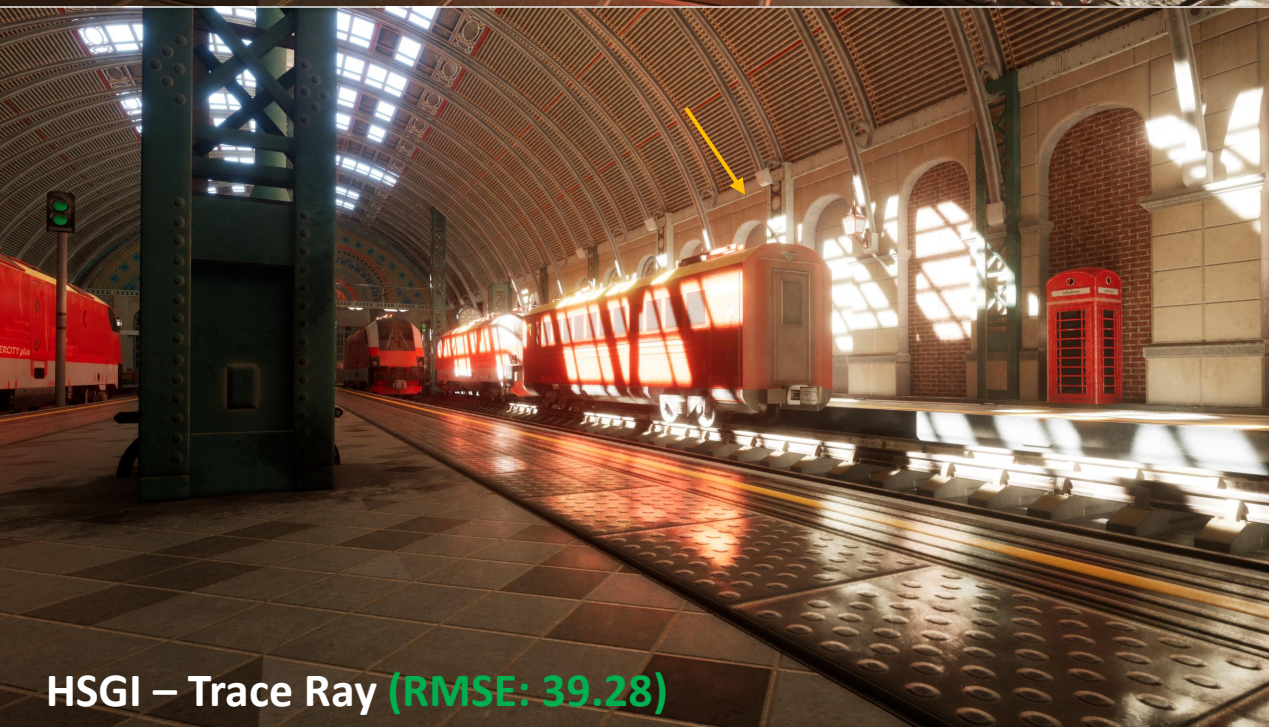


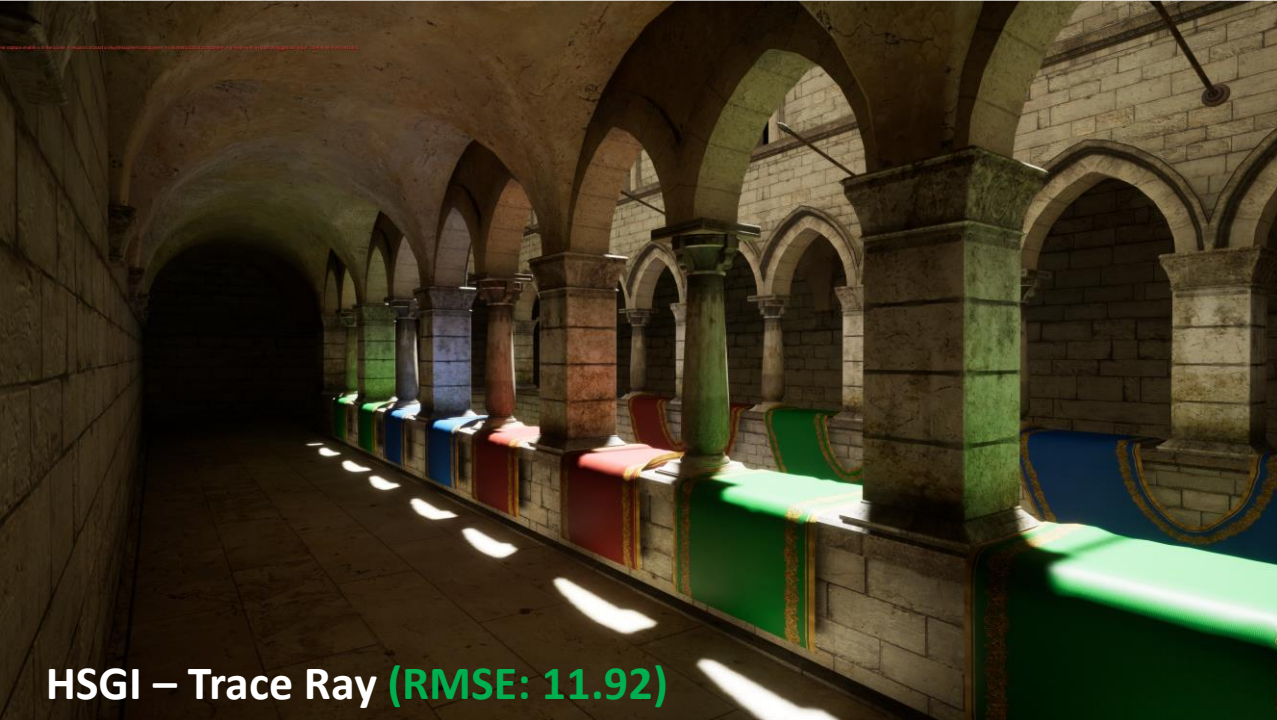
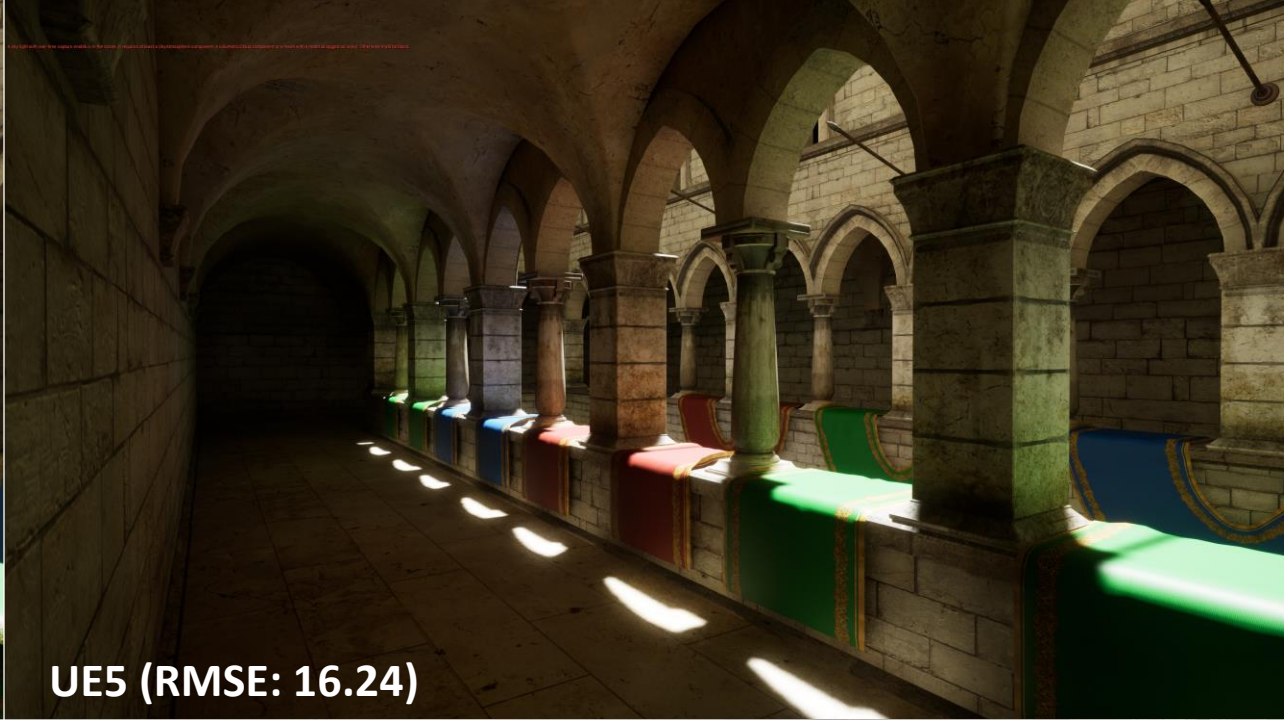
Show Cases











Demo Video



Summary

- **Fully dynamic game scene**
 - Support geometry animation and destruction at large scale
- **PC/Console**
 - Trace ray mode & trace probe mode
 - Quality & stability
 - ReSTIR GI Shadow
 - Retain indirect lighting shadow
 - Approach similar quality to path tracing
- **Mobile**
 - Deferred screen probe
 - Optimize DDGI to > 60fps
- **No HWRT dependency**





THANKS

March 20-24, 2023 | San Francisco, CA

Future Work

- **PC/Console**

- Improve convergence with world space sampling (ReGIR)
- Further Reduce bias for specular reflection and diffuse probe trace
- Suppress color noise
- One more bounce for specular reflection
- More BSDF support
- Translucent Voxel
- AI based radiance cache and denoiser

- **Mobile**

- Bent normal AO
- Hit point filtering reflection
- Fast probe convergence speed
- Low probe energy bias



References

- [BundleFusion 2017] "BundleFusion: Real-time Globally Consistent 3D Reconstruction using On-the-fly Surface Re-integration", TOG 2017
- [GVDB 2016] "GVDB: Raytracing Sparse Voxel Database Structures on the GPU", HG 2016
- [DDGI 2019] "Dynamic Diffuse Global Illumination with Ray-Traced Irradiance Fields", JCGT 2019
- [Schied 2017] "Spatiotemporal variance-guided filtering: real-time reconstruction for path-traced global illumination", HPG 2017
- [Bitterli 2020] "Spatiotemporal reservoir resampling for real-time ray tracing with dynamic direct lighting", TOG 2020
- [Ouyang 2021] "ReSTIR GI: Path resampling for real-time path tracing", CGF 2021
- [Wyman 2021] "Rearchitecting spatiotemporal resampling for production", HPG 2021
- [Lin 2022] "Generalized resampled importance sampling: foundations of ReSTIR", TOG 2022
- [Boksansky] "Rendering Many Lights with Grid-Based Reservoirs", Ray Tracing Gems II 2021
- [Boissé 2021] "World-space spatiotemporal reservoir reuse for ray-traced global illumination", SIGGRAPH Asia 2021
- [Stachowiak 2015] "Stochastic Screen-Space Reflections", SIGGRAPH Courses 2015
- [Stachowiak 2018] "Stochastic All The Things: Raytracing in Hybrid Real-Time Rendering", Digital Dragons 2018
- [Uludag 2014] "Hi-Z Screen-Space Cone-Traced Reflections", GPU Pro 5
- [Wright 2022] "Lumen: Real-time Global Illumination in Unreal Engine 5", SIGGRAPH Courses 2022
- [Zhdan 2021] "ReBLUR: A Hierarchical Recurrent Denoiser", Ray Tracing Gems II 2021
- [NRD] "NVIDIA Real-time Denoisers", <https://github.com/NVIDIAGameWorks/RayTracingDenoiser/>
- [Kajiya] "Experimental real-time global illumination renderer made with Rust and Vulkan", <https://github.com/EmbarkStudios/kajiya>

