Rendering
UE 5 Lumen

CS 415: Game Development

Professor Eric Shaffer

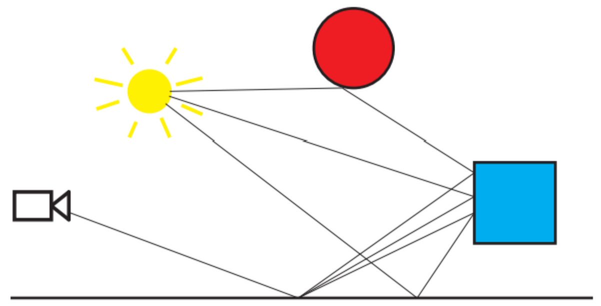
Material based on the SIGGRPAH 2022 Presentation
Lumen: Real-time Global Illumination in *Unreal Engine 5*
Daniel Wright, Krzysztof Narkowicz, Patrick Kelly

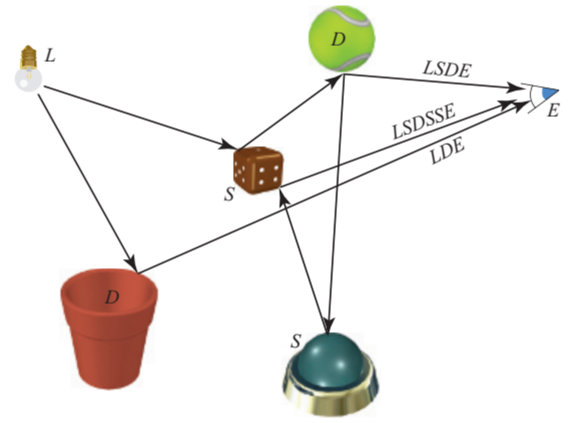
Global Illumination

"Global illumination…is a group of algorithms used in 3D computer graphics that are meant to add more realistic lighting to 3D scenes. Such algorithms take into account not only the light that comes directly from a light source (direct illumination), but also subsequent cases in which light rays from the same source are reflected by other surfaces in the scene, whether reflective or not (indirect illumination).

Theoretically, reflections, refractions, and shadows are all examples of global illumination, because when simulating them, one object affects the rendering of another (as opposed to an object being affected only by a direct source of light). In practice, however, only the simulation of diffuse inter-reflection or caustics is called global illumination.*" -- Wikipedia

* Not really true



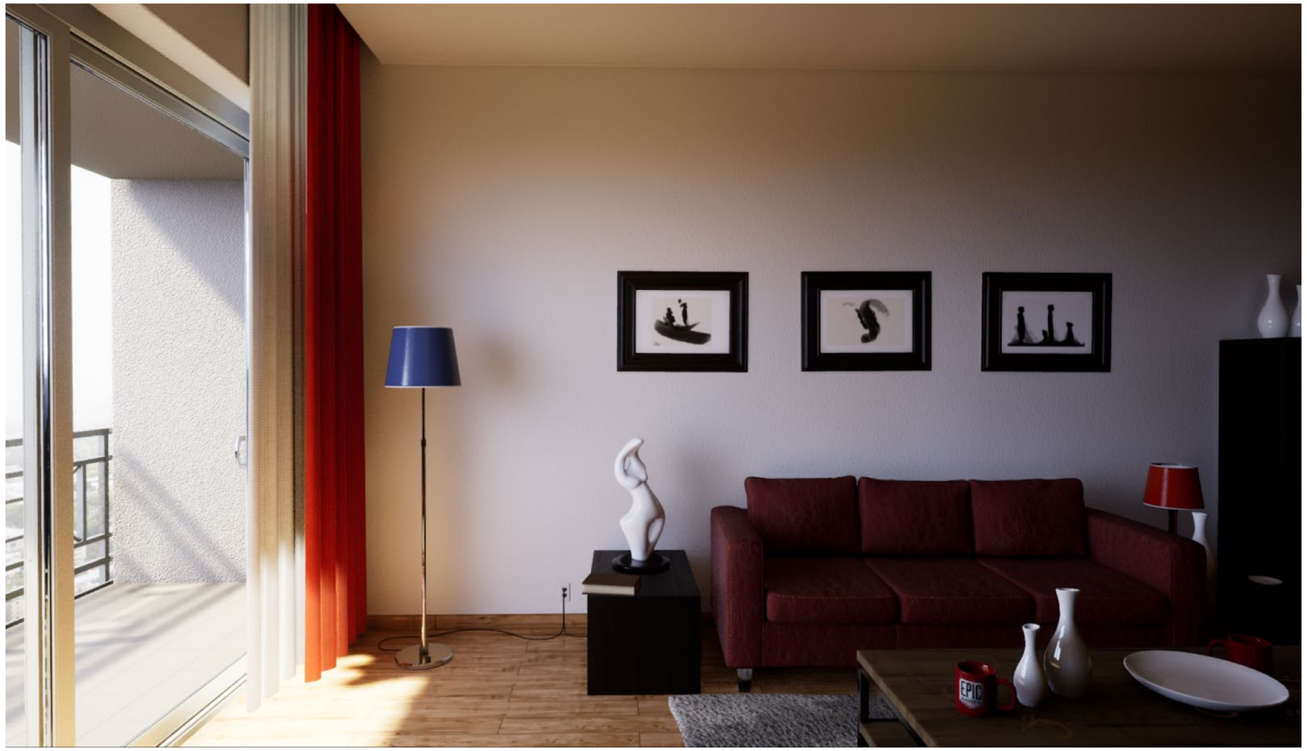


ILLINOIS

Lumen

Lumen is Unreal Engine 5's fully dynamic global illumination and reflections system

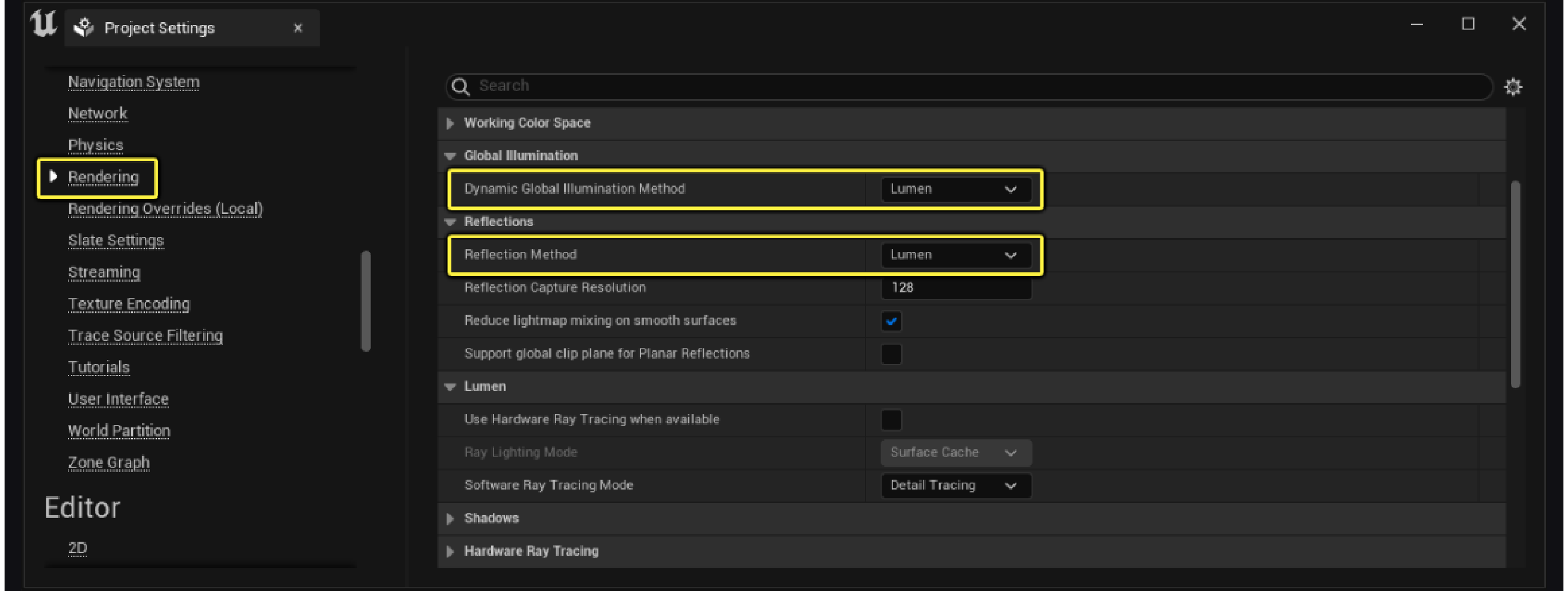
- Enabled out of the box.
- Designed for next-generation consoles
- Supports high-end visualizations like architectural visualization



“Lumen provides infinite(?) diffuse bounces, which are important in scenes with bright diffuse materials like the white paint in the apartment in the image”

Lumen is The Default GI Method in UE5

Lumen is enabled from the Project Settings under the **Rendering > Dynamic Global Illumination** and **Reflections** categories.



Working Color Space

Global Illumination

Dynamic Global Illumination Method	Lumen

Reflections

Reflection Method	Lumen
Reflection Capture Resolution	128
Reduce lightmap mixing on smooth surfaces	✓
Support global clip plane for Planar Reflections	

Lumen

Use Hardware Ray Tracing when available	
Ray Lighting Mode	Surface Cache
Software Ray Tracing Mode	Detail Tracing

Shadows

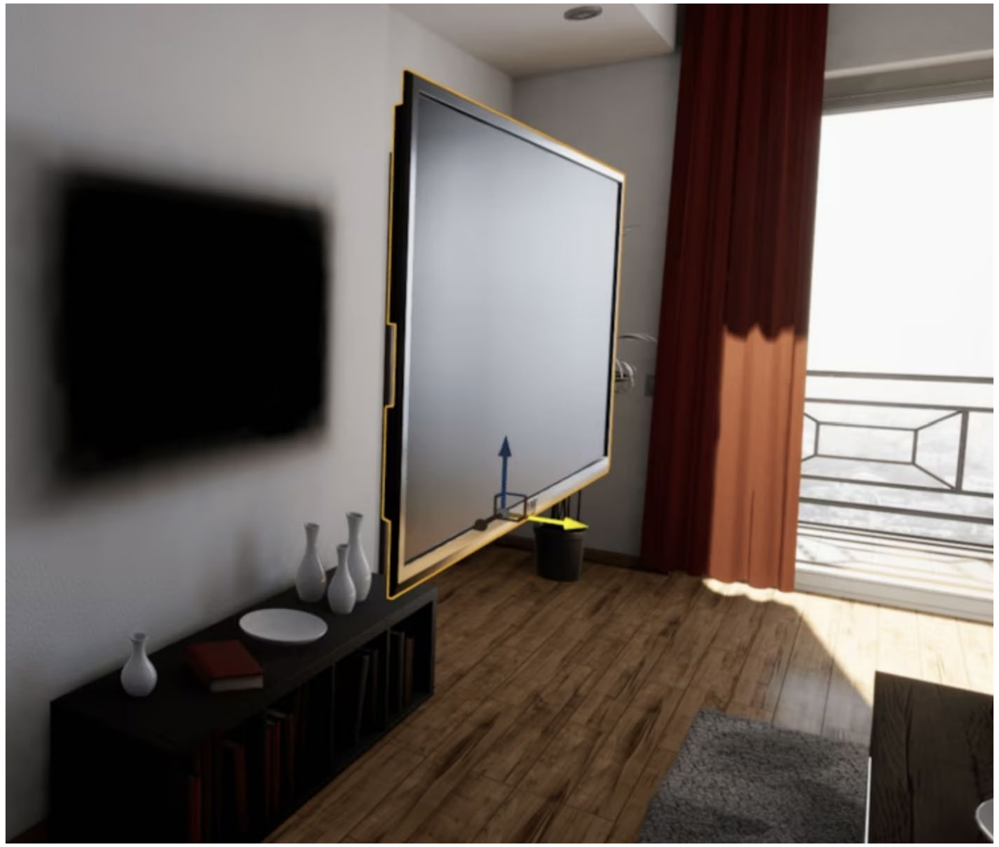
Hardware Ray Tracing

Global Illumination and Reflections can be set independently. In each category, set the following to enable Lumen:

- Dynamic Global Illumination: **Lumen**
- Reflection Method: **Lumen**

Previously…

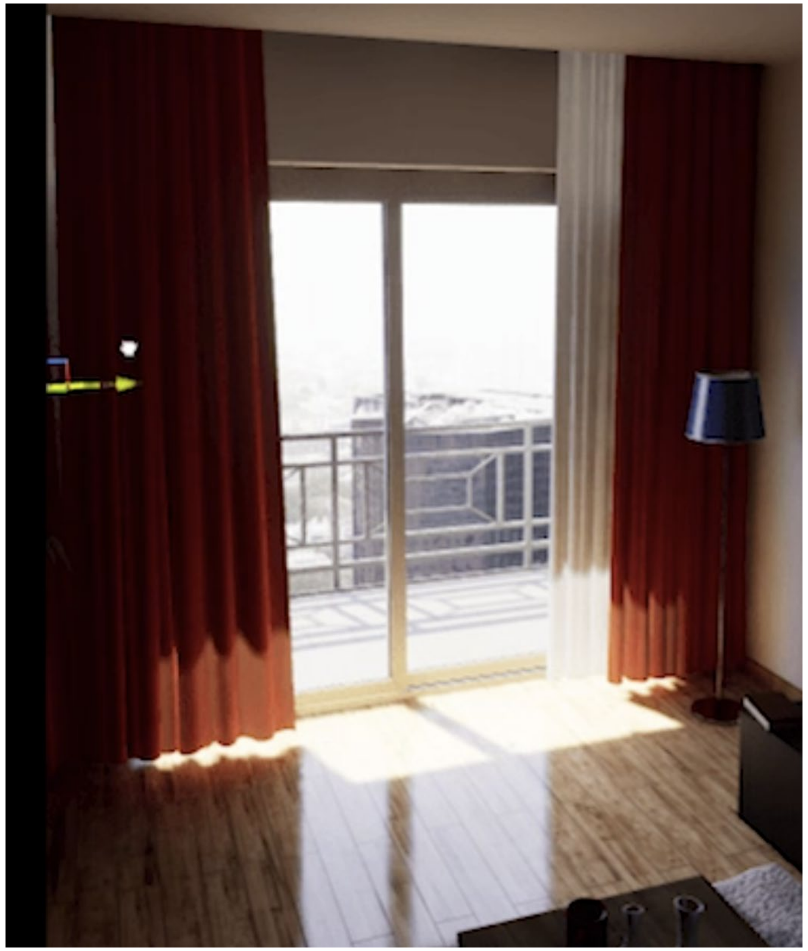
“In the past, global illumination, for most games, had to be solved in an offline process called **lightmap baking**, because it was too computationally expensive to be calculated in real time. In Unreal Engine, lightmaps are baked through CPU Lightmass or GPU Lightmass. Static lighting from lightmaps can provide very high quality, but requires long build times, and greatly constrains the game environment. Any action which changes the indirect lighting in a significant way, like moving a wall-mounted television, will leave lighting in an incorrect state.”



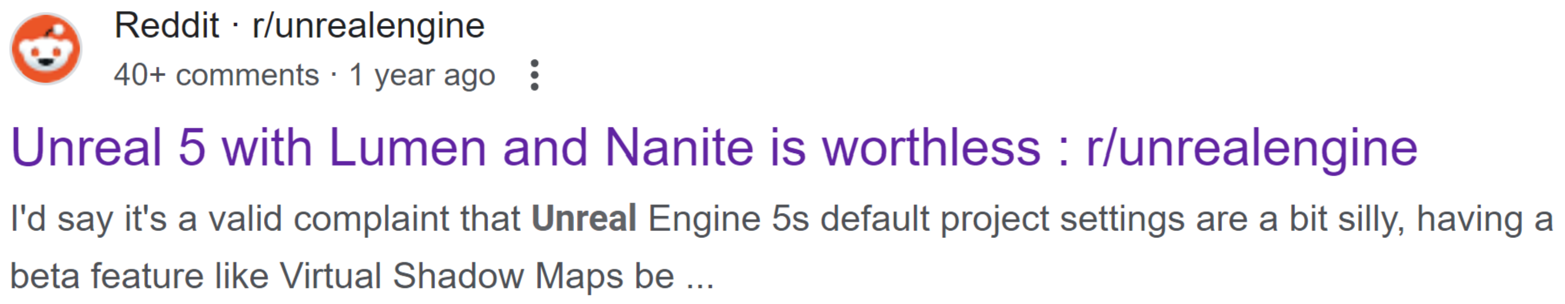
Previously…

“Games that required these dynamic scenes have been relying on low-quality irradiance probe-based lighting, and approximations like ambient occlusion.

Lumen simulates light bouncing around the scene in real time, enabling players to change any aspect of the game world, with indirect lighting automatically updating. Players can destroy large parts of levels, change the time of day, or flood a portion of the level and those changes will automatically propagate to the lighting. Game developers can say goodbye to seeing the ‘LIGHTING NEEDS TO BE REBUILT’ message in the Unreal Editor.”



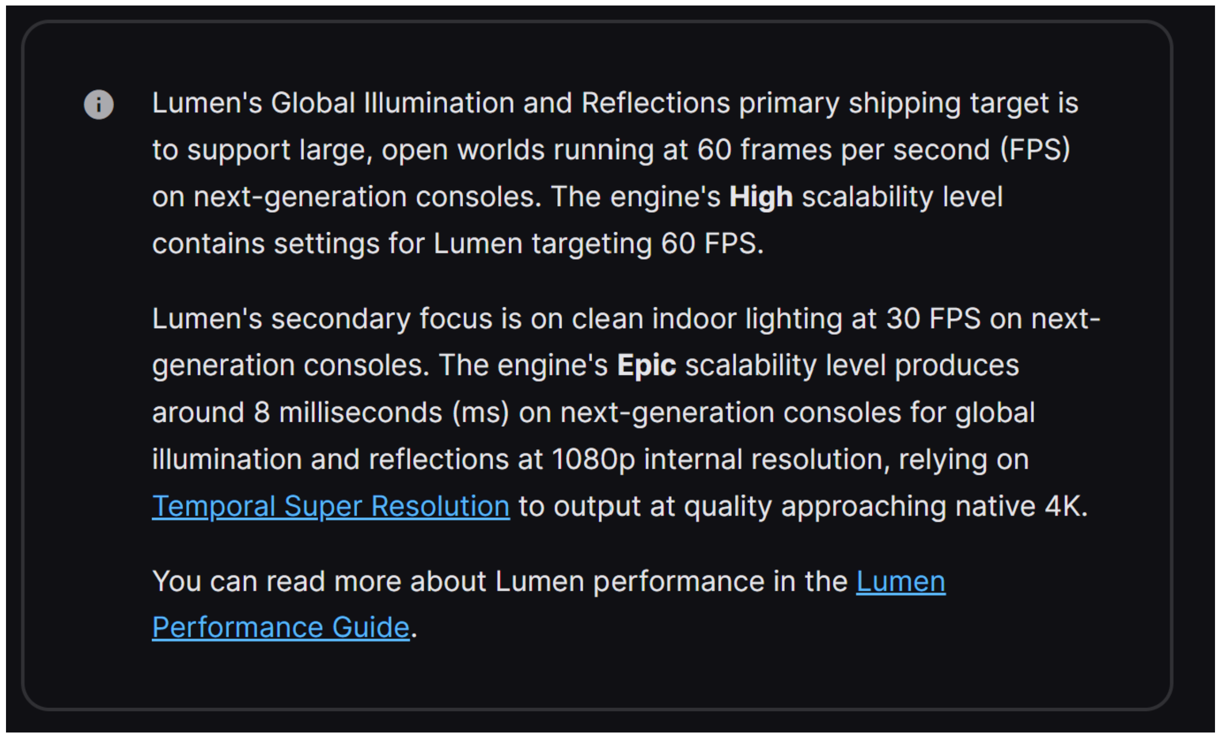
Understanding the Target Applications and Performance



Reddit · r/unrealengine
40+ comments · 1 year ago

Unreal 5 with Lumen and Nanite is worthless : r/unrealengine

I'd say it's a valid complaint that **Unreal** Engine 5s default project settings are a bit silly, having a beta feature like Virtual Shadow Maps be ...



Lumen's Global Illumination and Reflections primary shipping target is to support large, open worlds running at 60 frames per second (FPS) on next-generation consoles. The engine's **High** scalability level contains settings for Lumen targeting 60 FPS.

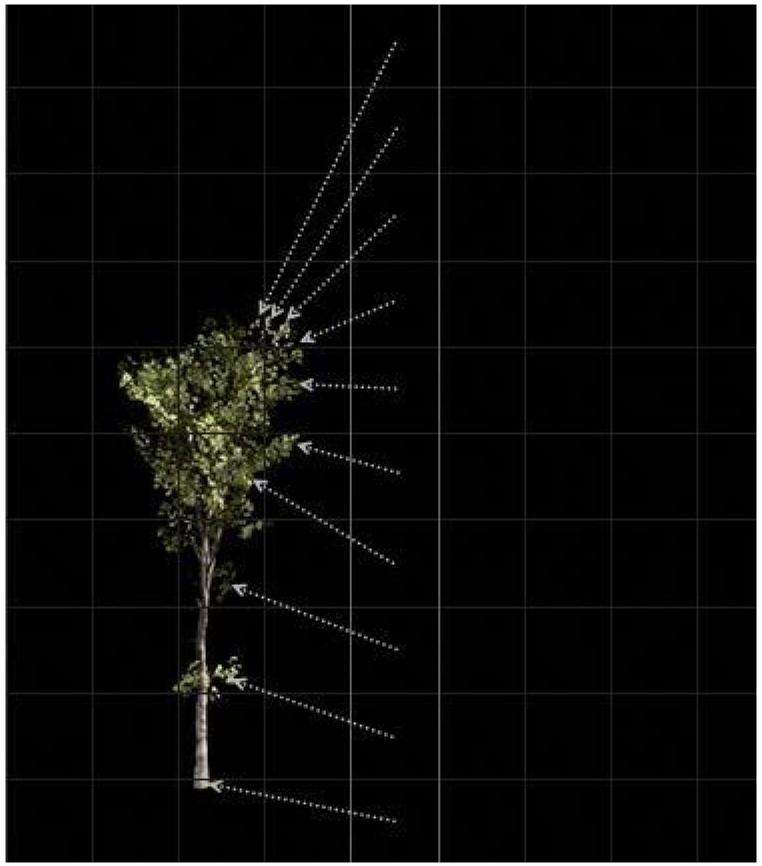
Lumen's secondary focus is on clean indoor lighting at 30 FPS on next-generation consoles. The engine's **Epic** scalability level produces around 8 milliseconds (ms) on next-generation consoles for global illumination and reflections at 1080p internal resolution, relying on Temporal Super Resolution to output at quality approaching native 4K.

You can read more about Lumen performance in the Lumen Performance Guide.

So…What Does Lumen Do?

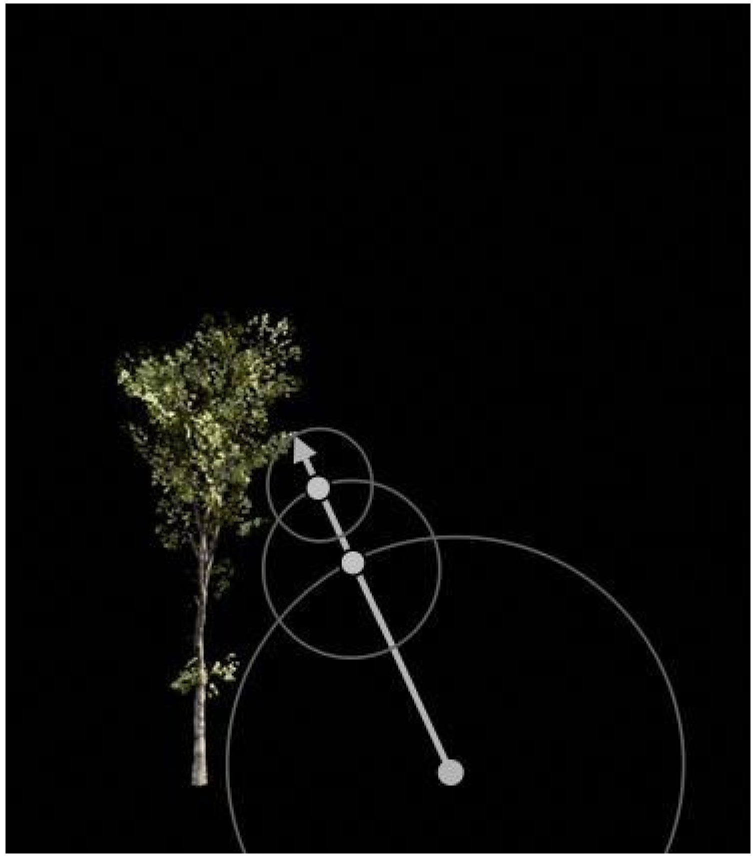
"Lumen uses multiple ray-tracing methods to solve Global Illumination and Reflections. Screen Traces are done first, followed by a more reliable method. Lumen uses Software Ray Tracing through Signed Distance Fields by default, but can achieve higher quality on supporting video cards when Hardware Ray Tracing is enabled."

What is a Signed Distance Field?



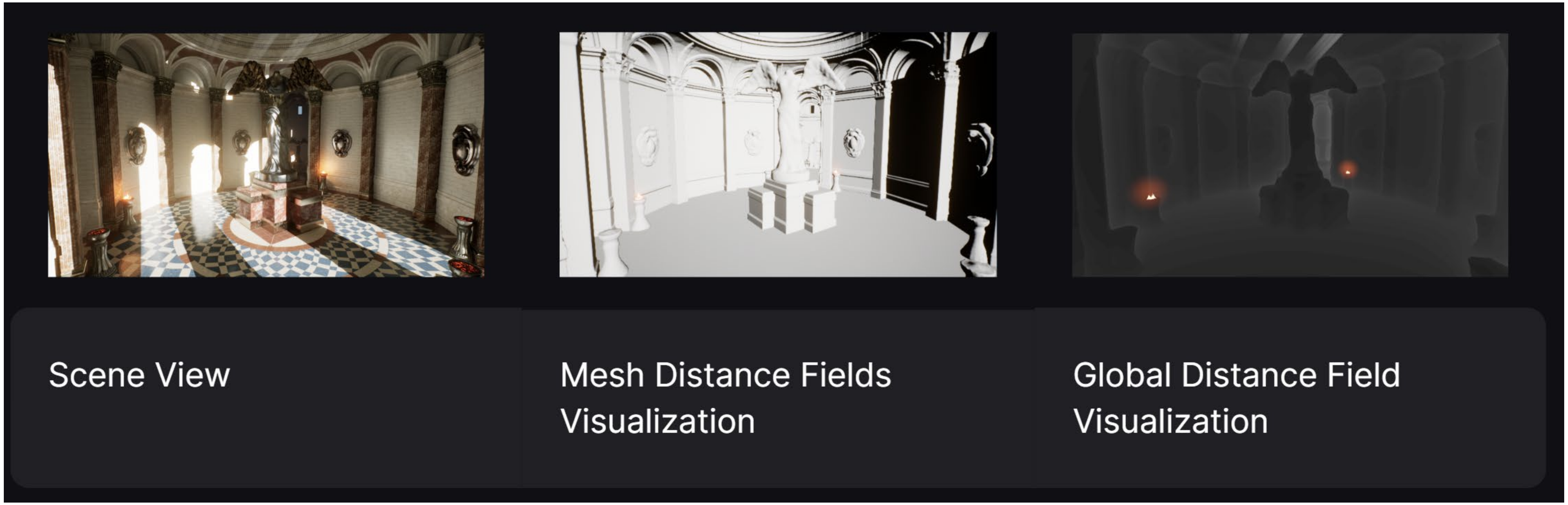
To represent a Static Mesh's surfaces, a **Signed Distance Field** (SDF) is used. It works by storing the distance to the nearest surface in a volume texture. For every point on the exterior of the mesh is considered positive distance and any point inside the mesh stores a negative distance. In the example below, the positive distances are being traced and stored to represent the tree later on.

Ray Tracing SDFs



When tracing a ray, you can safely skip through empty space as the distance to the nearest surface is already known (this is sometimes called Sphere Tracing). This allows the intersections to be determined with a small number steps. By ray tracing a distance field, a visibility result is produced, meaning that if the ray intersects the mesh, the light will then be shadowed.

SDF Constructions

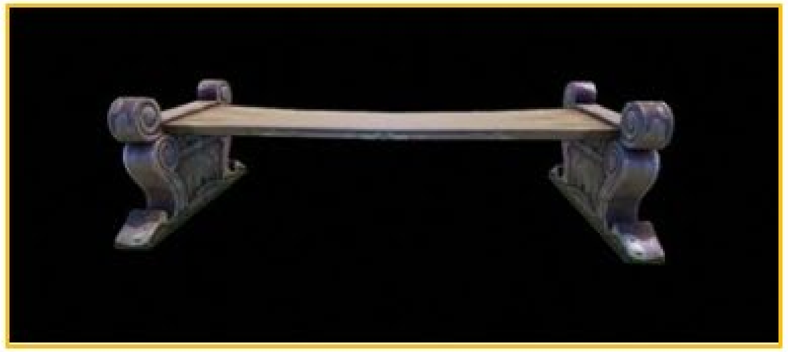

Scene View
Mesh Distance Fields Visualization
Global Distance Field Visualization

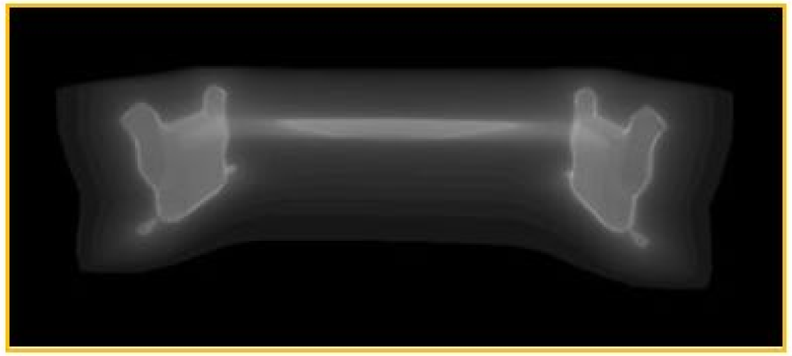
Individual Static Meshes have precomputed SDFs

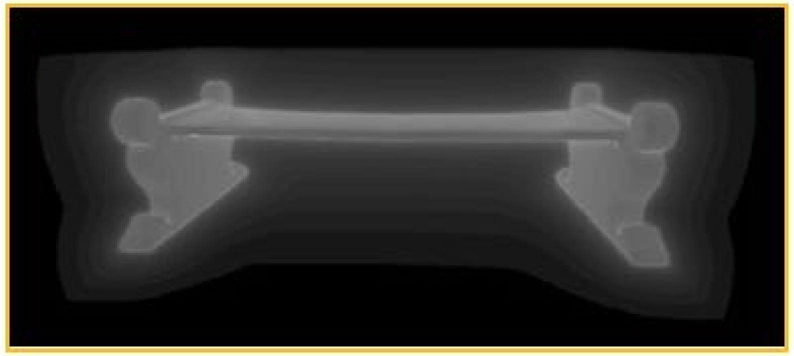
These are merged into a Global Distance Field for the Scene

ILLINOIS

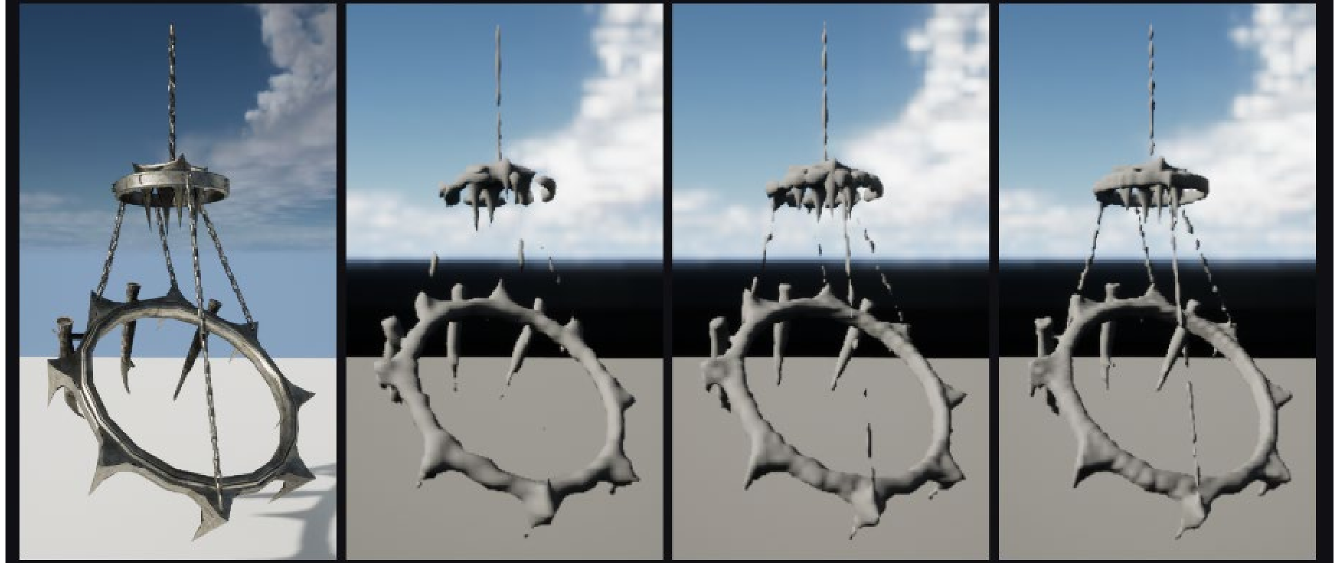
There Can Be Problems….


Original Mesh


Resolution is too low, important features are lost.

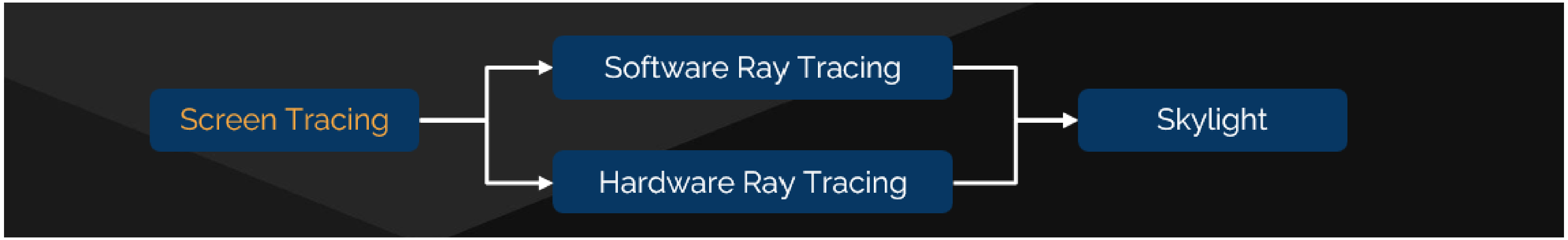

Resolution has been increased, important features represented

For some meshes, thin surfaces may not have a good distance field representation and could cause issues with light leaking. The Mesh Distance Field visualization can help you spot these types of issues.


(Left to Right) Triangle Mesh, Distance Field Resolution Scales 1.0 (Default), 1.5, 2.0

Can require lots of manual tweaking of scenes and models to get high quality rendering and high performance

Lumen System Architecture



Lumen provides two methods of ray tracing the scene:

Software Ray Tracing: uses Mesh Distance Fields to operate on the widest range of hardware and platforms but is limited in the types of geometry, materials, and workflows it can effectively use.

Hardware Ray Tracing: supports a larger range of geometry types for high quality by tracing against triangles and to evaluate lighting at the ray hit instead of the lower quality Surface Cache. It requires supported video cards and systems to operate.

Software Ray Tracing is the only performant option in scenes with many overlapping instances, while Hardware Ray Tracing is the only way to achieve high quality mirror reflections on surfaces.

Challenges

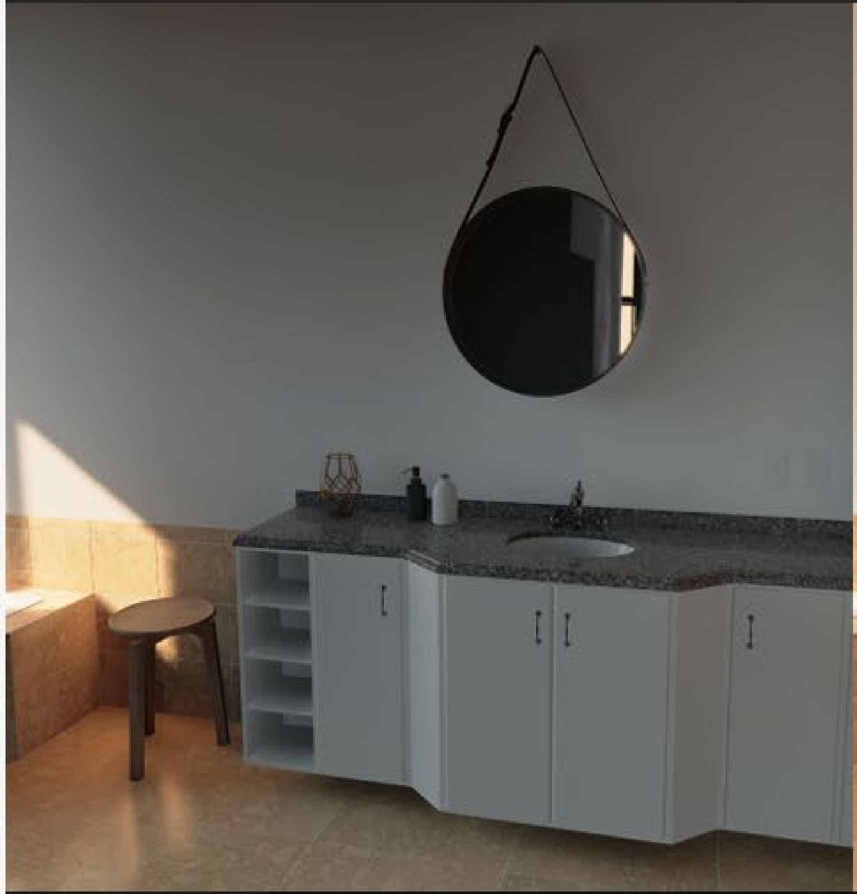
- Lighting build has 100,000x more processing time, even in small maps
- Mapping incoherent light transfer to GPU's efficiently
- Huge problem space
- Razor's edge balance between performance and quality

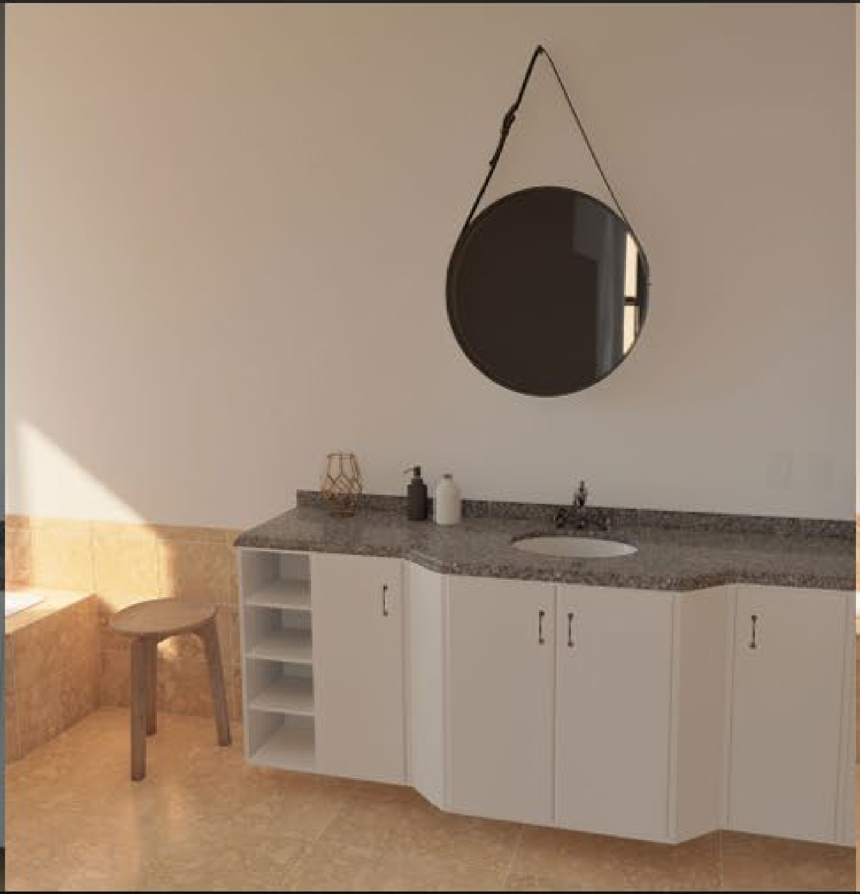
Let's look at a few slides from the SIGGRAPH research talk to understand the goals and issues the engineers addressed…

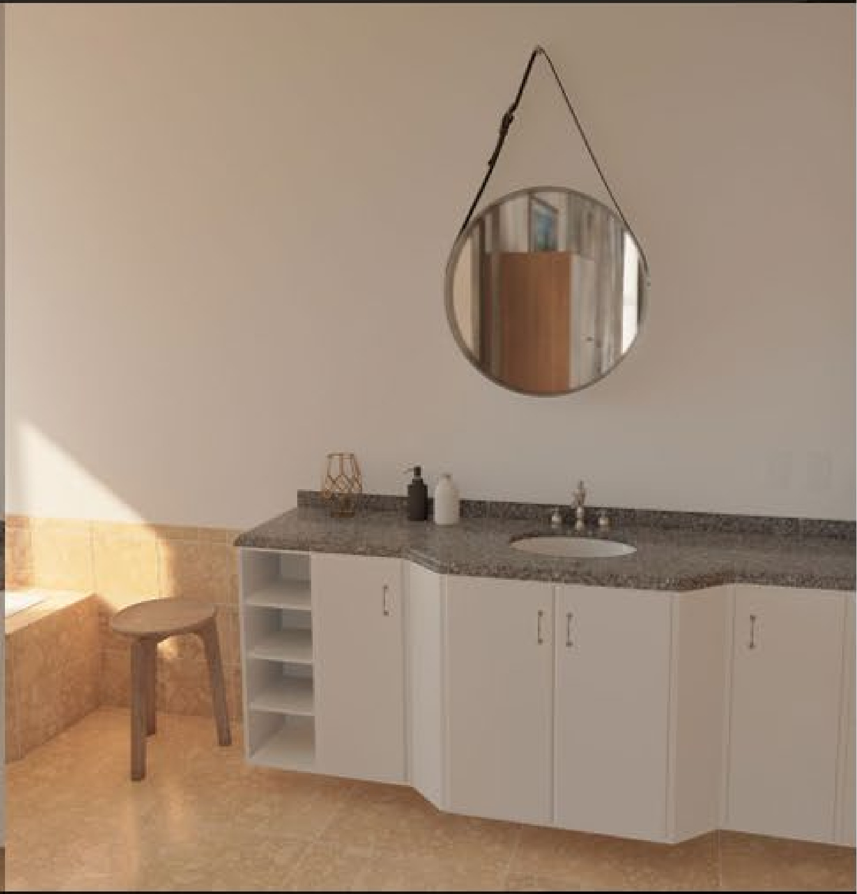
Fundamental problem #1:
How to trace rays?

- Hardware Ray Tracing is great, but
  - Need options to scale down
    - PC market, 60fps console
  - And handle heavily overlapping meshes that are slow with HWRT
- Also want a Software Ray Tracing path that addresses these

Fundamental problem #2:
How to solve the whole indirect lighting path?

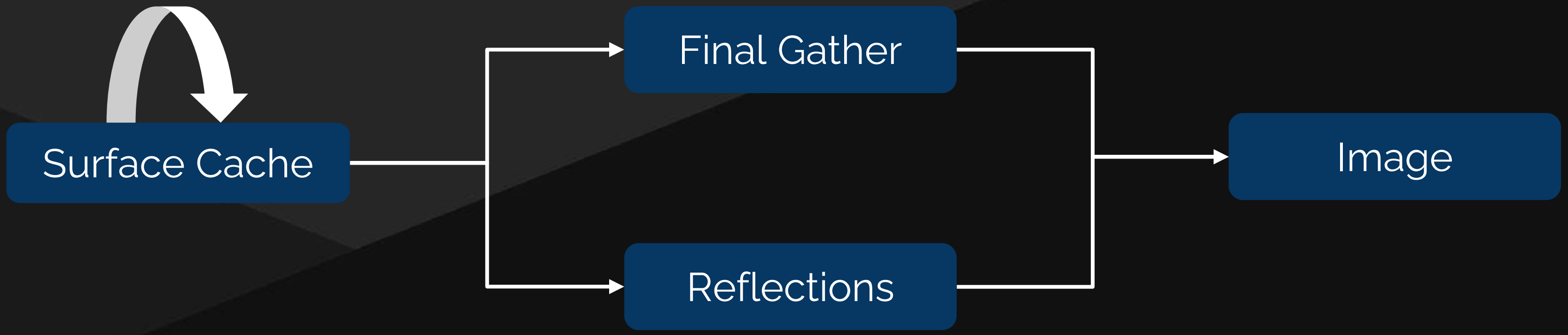

Single bounce


Multi-bounce diffuse


GI seen in reflections

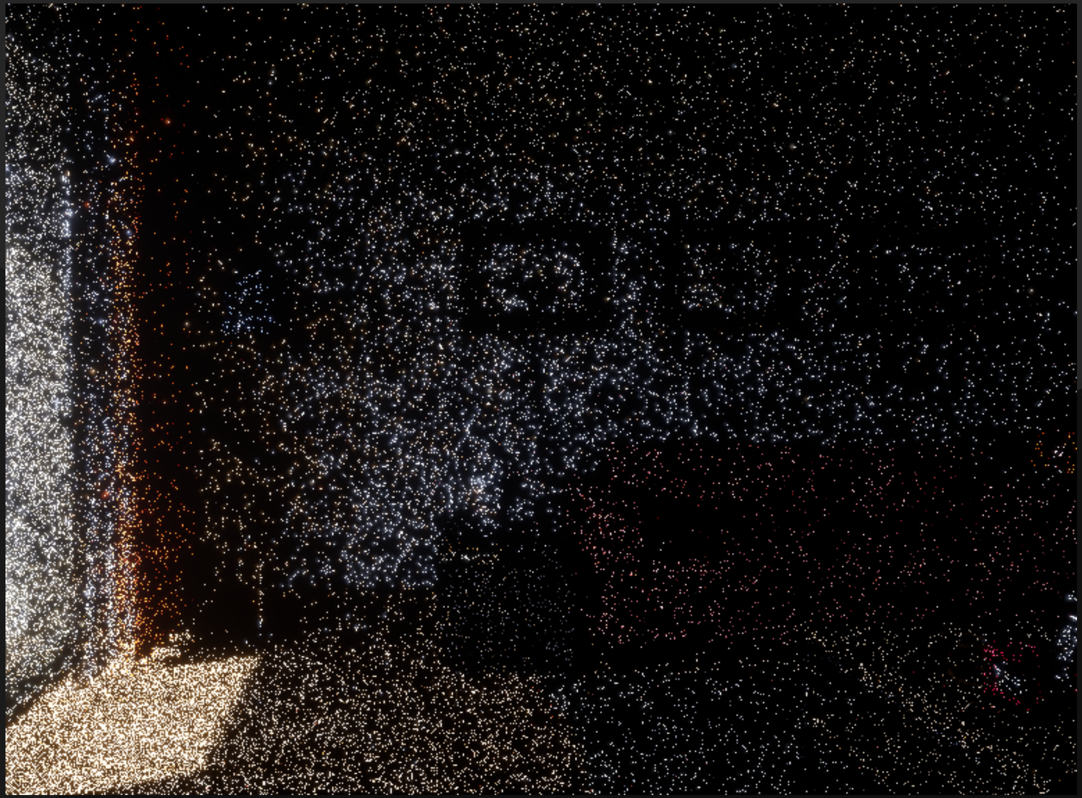
First bounce is the most important, needs careful sampling

- Final Gather (diffuse)
- Reflection denoising (specular)
- Multi-bounce diffuse through Surface Cache feedback

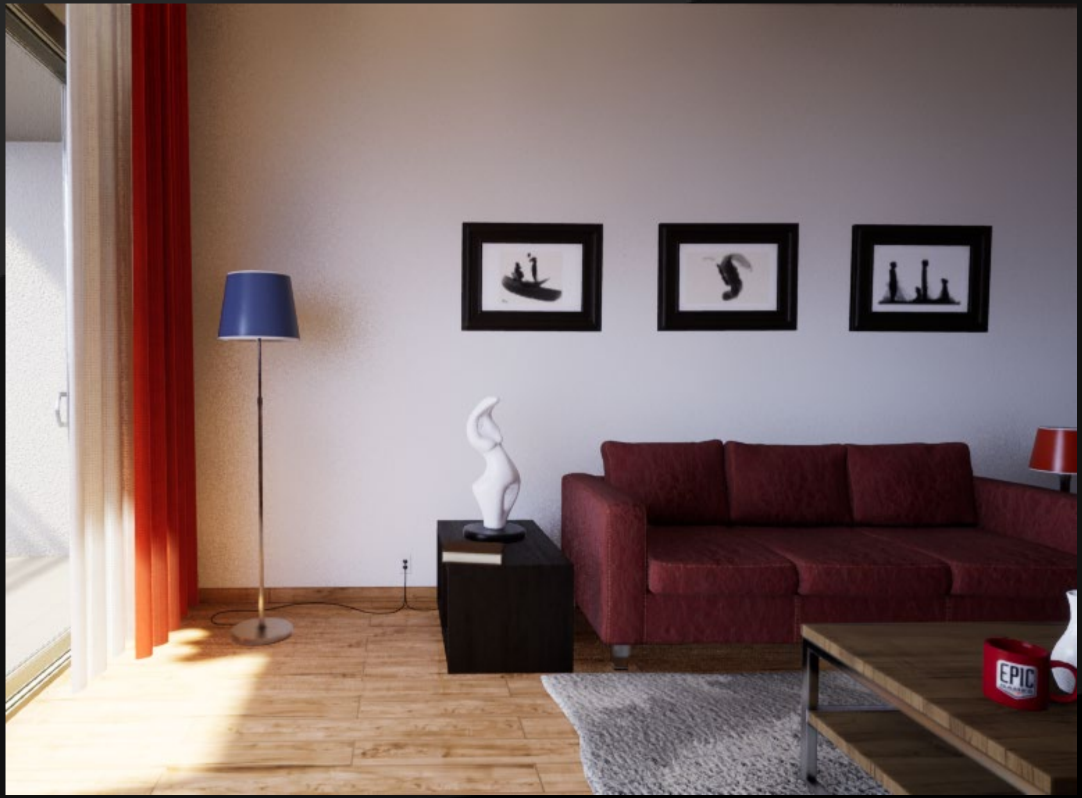


Fundamental problem #3:
How to solve noise in the light transfer?

- We can't even afford one ray per pixel
  - But need hundreds of effective samples for high quality indoor GI



$\xrightarrow{?}$

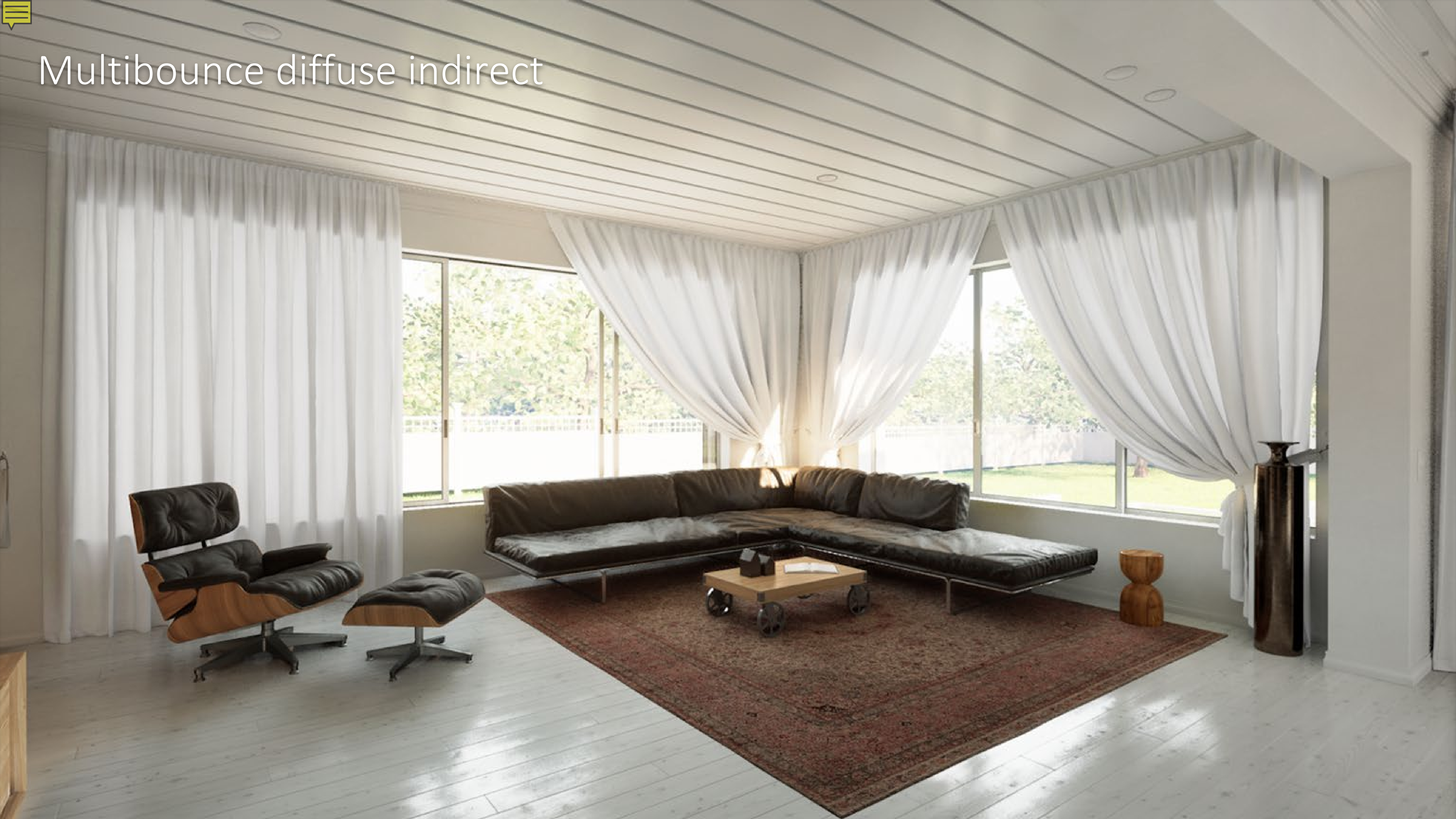


Final Gather techniques

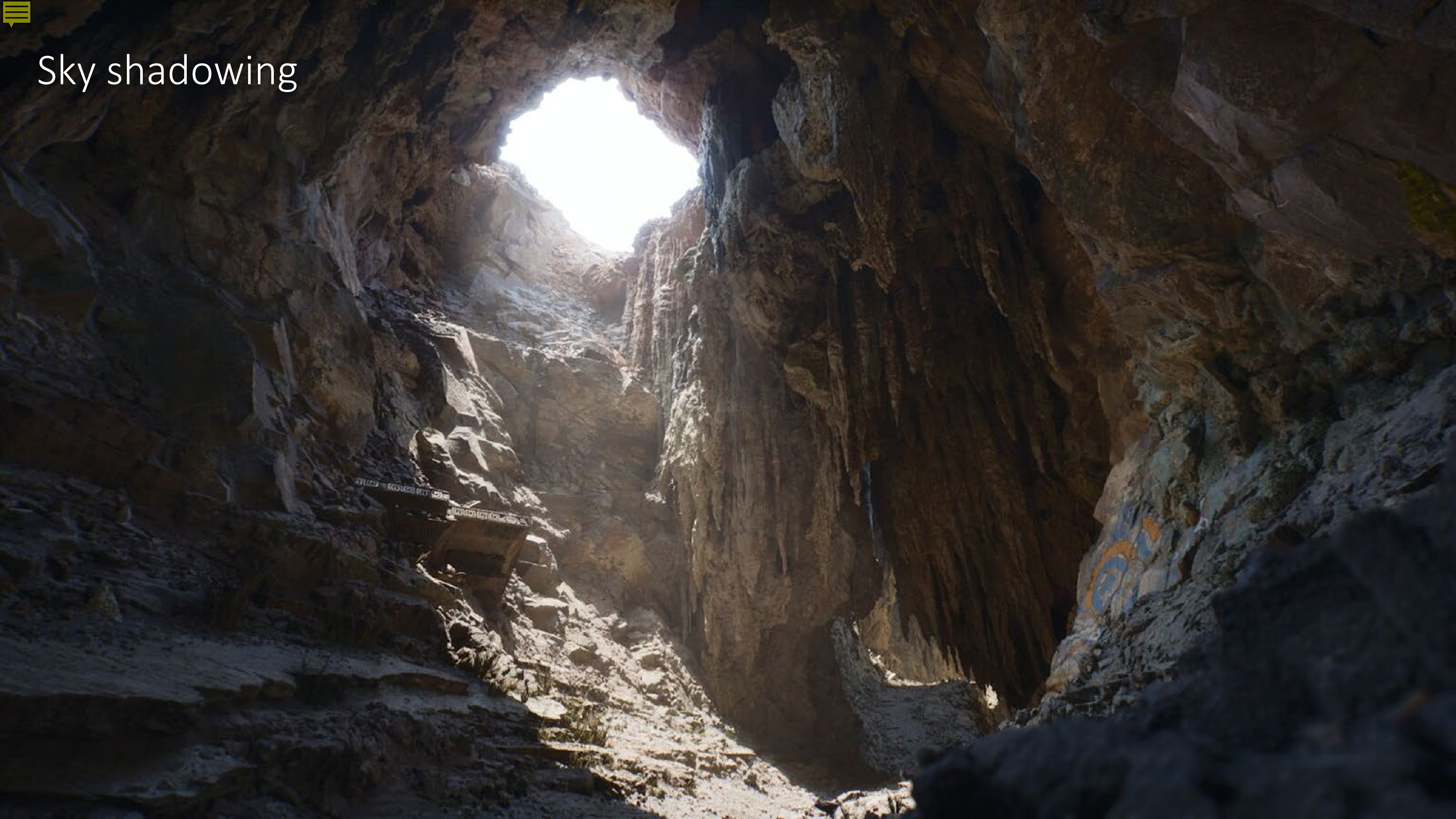
- Adaptive downsampling
- Spatial and temporal reuse
- Product Importance Sampling

Enough of the research…let's look at some results….

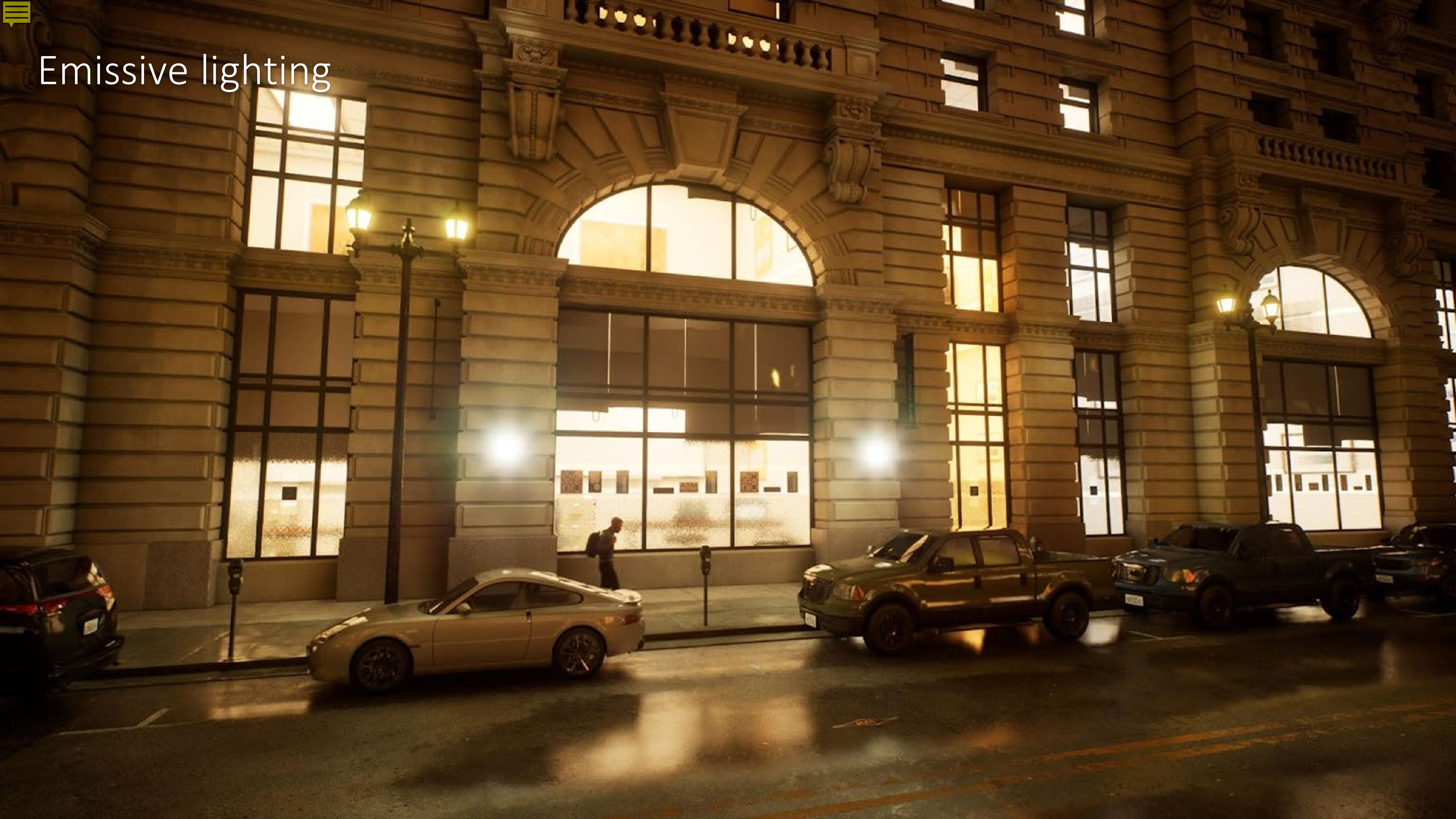
Multibounce diffuse indirect



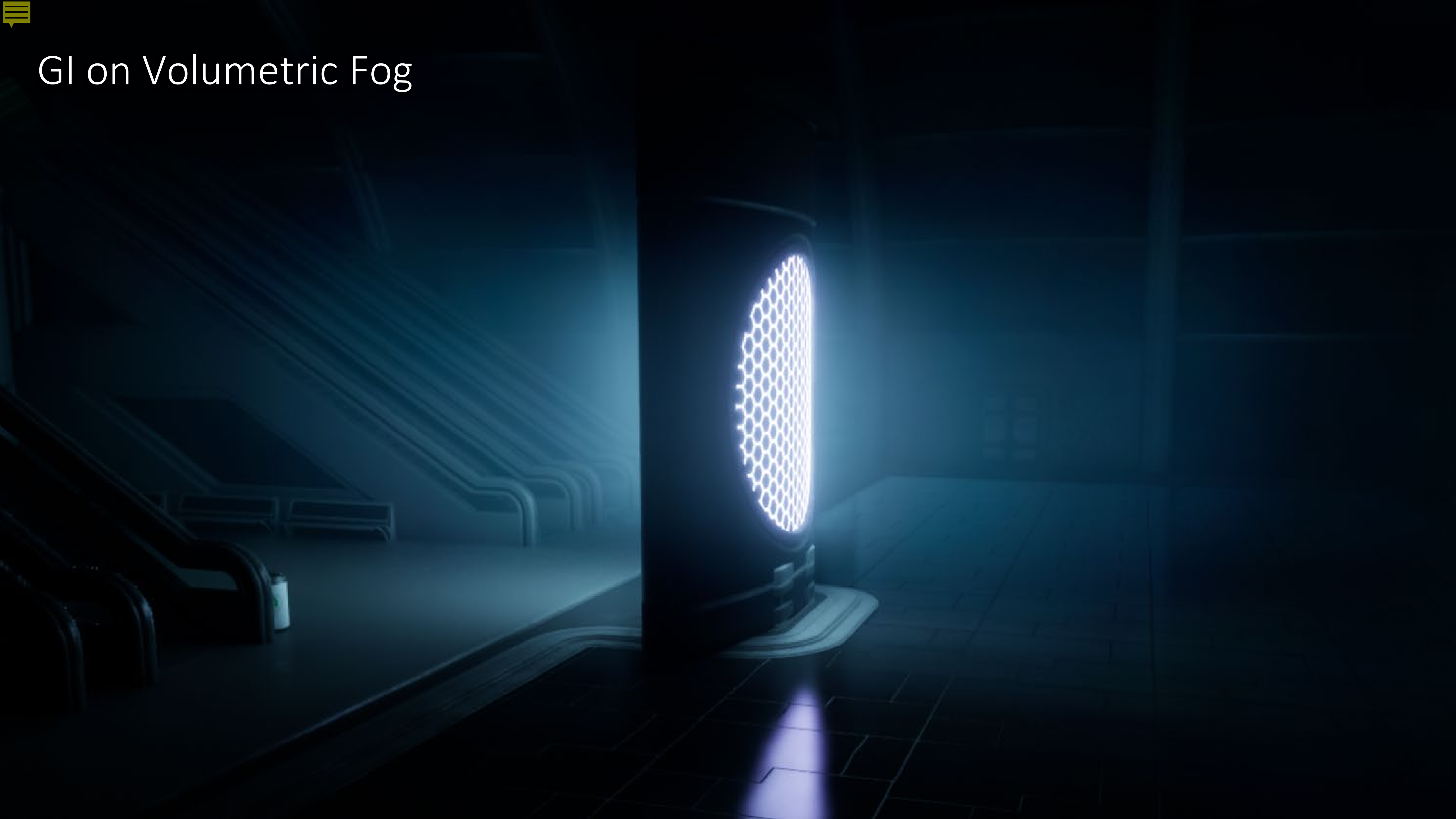
Sky shadowing

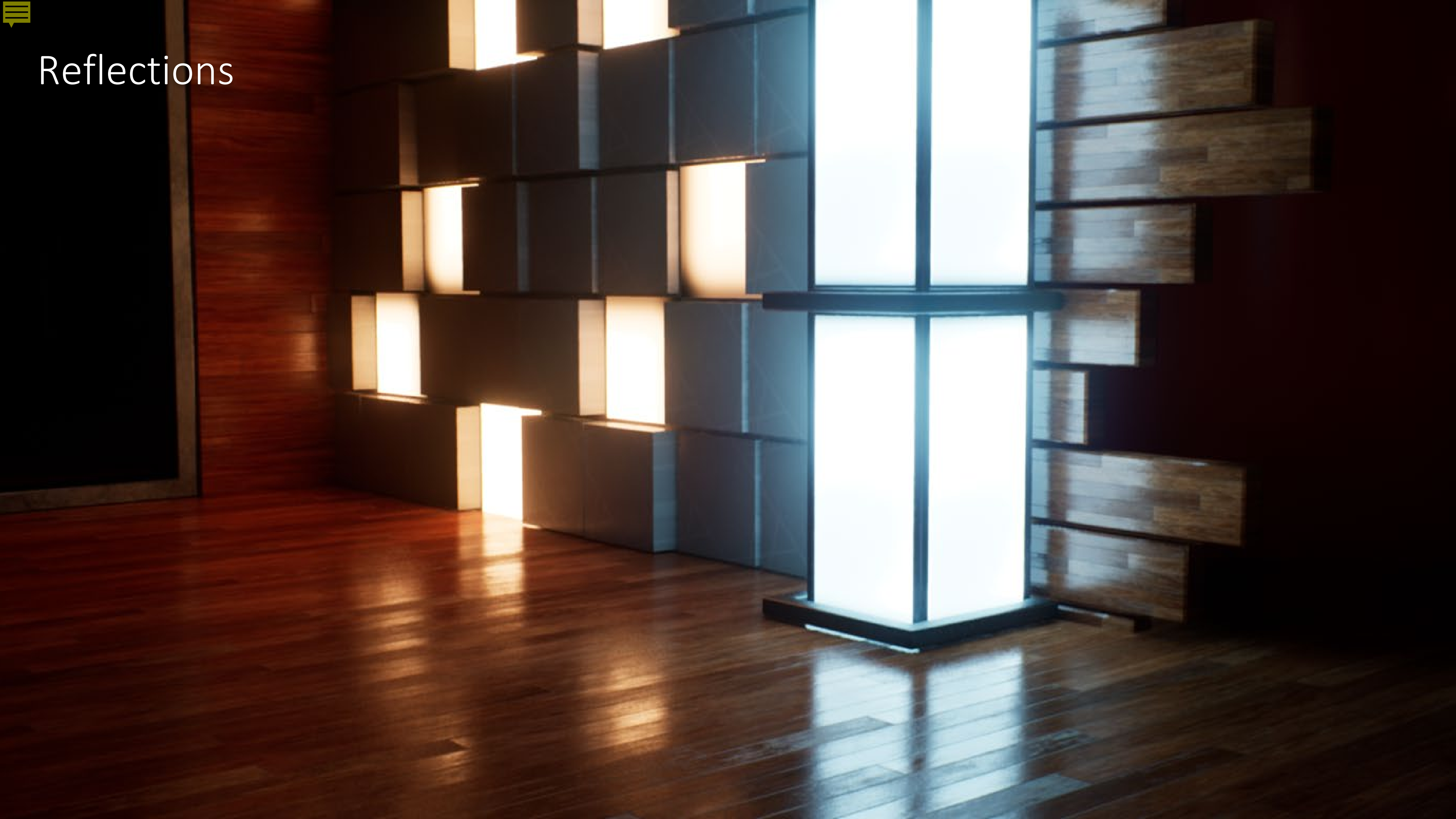


Emissive lighting



GI on Volumetric Fog





Reflections