Advanced Lighting Techniques in Unity

Unity DevDay GDC’14
Jesper Mortensen & Kuba Cupisz (in absentia) – Unity Technologies

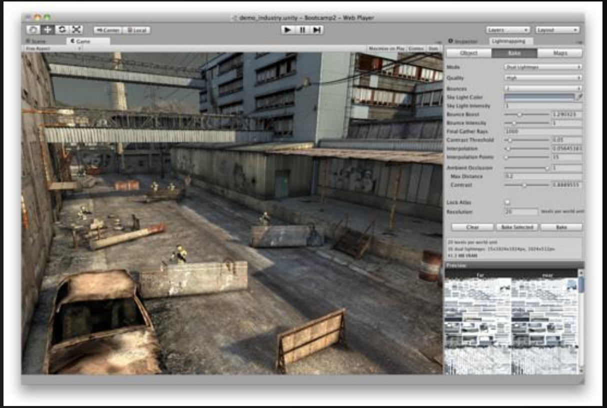
Presented by:
Aras Pranckevičius – Unity Technologies
Jens Fursund – Imagination Technologies
Sam Martin – Geomerics

Talk overview

- Current state of play in Unity
- The ideal lighting solution
- New Unity 5.0 Editor workflow
- New Unity 5.0 dynamic Global Illumination runtime features
- Static Lightmap Baking Improvements in Unity 5.0
- Integration with Standard Shader in Unity 5.0
- The future…
- Questions

Current state of play in Unity

- Scope: Global illumination and advanced lighting
- Static lightmap baking
- Workflow for realistic lighting:
  - 1) setup scene
  - 2) lights
  - 3) press bake
  - 4) wait (have coffee or go home)
  - 5) fix problems go back to 1
- Not particularly nice for iteration…



Mode: Dual Lightmaps
Quality: High
Bounces: 1
Sky Light Color
Sky Light Intensity: 1
Bounce Boost: 1.290323
Bounce Intensity: 1
Final Gather Rays: 1000
Contrast Threshold: 0.05
Interpolation: 0.05645161
Interpolation Points: 15
Ambient Occlusion: 1
Max Distance: 0.2
Contrast: 1.888955
Lock Atlas
Resolution: 20 texels per world unit

20 texels per world unit
15 dual lightmaps: 15x1024x1024px, 1024x512px
41.3 MB VRAM

Current state of play in Unity

- Slow iteration means:
  - Fewer iteration cycles → lower quality
  - High cost for lighting
  - Frustration for developers/artists
- No dynamic runtime features available
  - Rules out time of day
  - Mood lighting
  - Gameplay features based on lighting

The ideal lighting solution

- Everything is dynamically lit, freely animate:
  - lights
  - materials
  - environment lighting
  - geometry
- Real-time performance on all platforms (mobile to high-end)
- No artifacts in the lighting
- “LOD” for lighting (ideally screen-space lighting)
- No extra authoring
- No precomputation time

The ideal lighting solution

- This lighting solution does not exist now
- Many engine providers have tried and failed
- Baking captures only a small subset of this ideal approach
- But there is a feasible approach that gets us most of the way
- Covers the majority of use-cases
- And is generally awesome…

New lighting solution in Unity 5.0

- Goals for the lighting solution:
  - Fast iteration cycle
  - Immediate feedback
  - Minimal authoring
  - Dynamic lighting including GI in-game
  - Available on a broad range of platforms, mobile and up

New lighting solution in Unity 5.0

- Enlighten lighting technology from Geomerics
  - Real-time global illumination
  - Dynamic lighting in-game and in-editor
  - High quality lightmap baking
  - Cross platform support (mobile to next-gen consoles)

- PowerVR Ray Tracing from Imagination Technologies
  - Interactive preview of GI in-editor
  - Unbiased physically based GI
  - Instant feedback on any change (including geometry changes)

New lighting solution in Unity 5.0

- The main new GI features are:
  - Dynamic indirect lighting
  - Dynamic materials (albedo, emissive)
  - Dynamic environment lighting
- Baking is still supported
  - Dynamic and baked lights can coexist
  - Objects are rendered with a seamless mix of baked and dynamic lights

New Unity 5.0 Editor workflow

- New iterative lighting workflow
- Quickly present accurate lighting (representative of the game)
- A background process tracks scene state
- Jobs are issued to fix up lighting using:
  - Threads
  - Processes
  - Cloud?

New Unity 5.0 Editor workflow

- Enlighten needs precompute time to prepare for real-time GI
- Meanwhile PowerVR fills in lighting using progressive path tracing
  - path tracing is initially noisy
  - but quickly converges
- PowerVR tuned to match Enlighten
- Both are:
  - unbiased
  - physically based

PowerVR Ray Tracing in Unity 5.0

Jens Fursund – Imagination Technologies

Interactive Light Maps in Unity 5.0

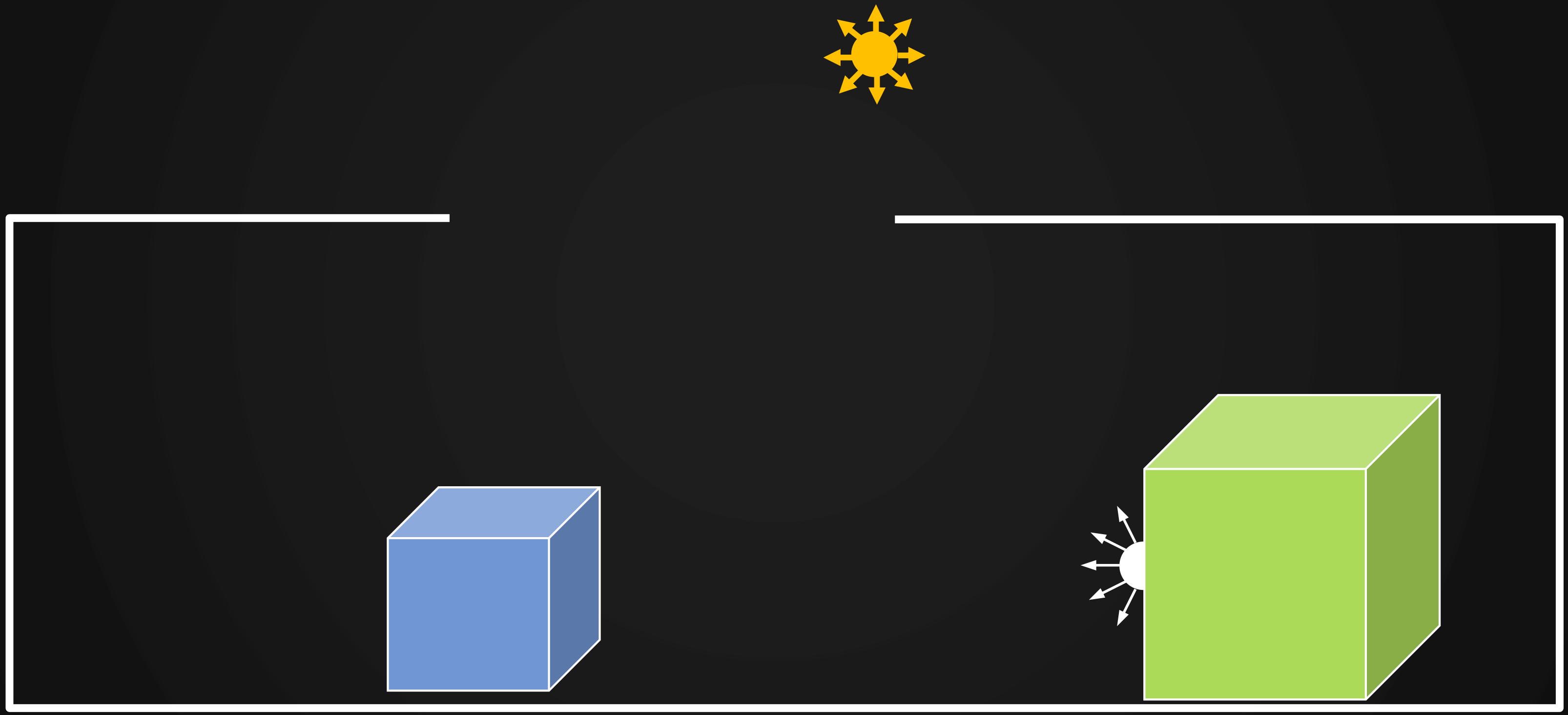
What does “Interactive” mean?

- A **lot** of work has been done to speed up the light map creation process
- Immediate feedback on lighting
- Uses same inputs as Enlighten
- Reliable results
- Allows for *fast* iteration
- Saves time and cost

DEMO

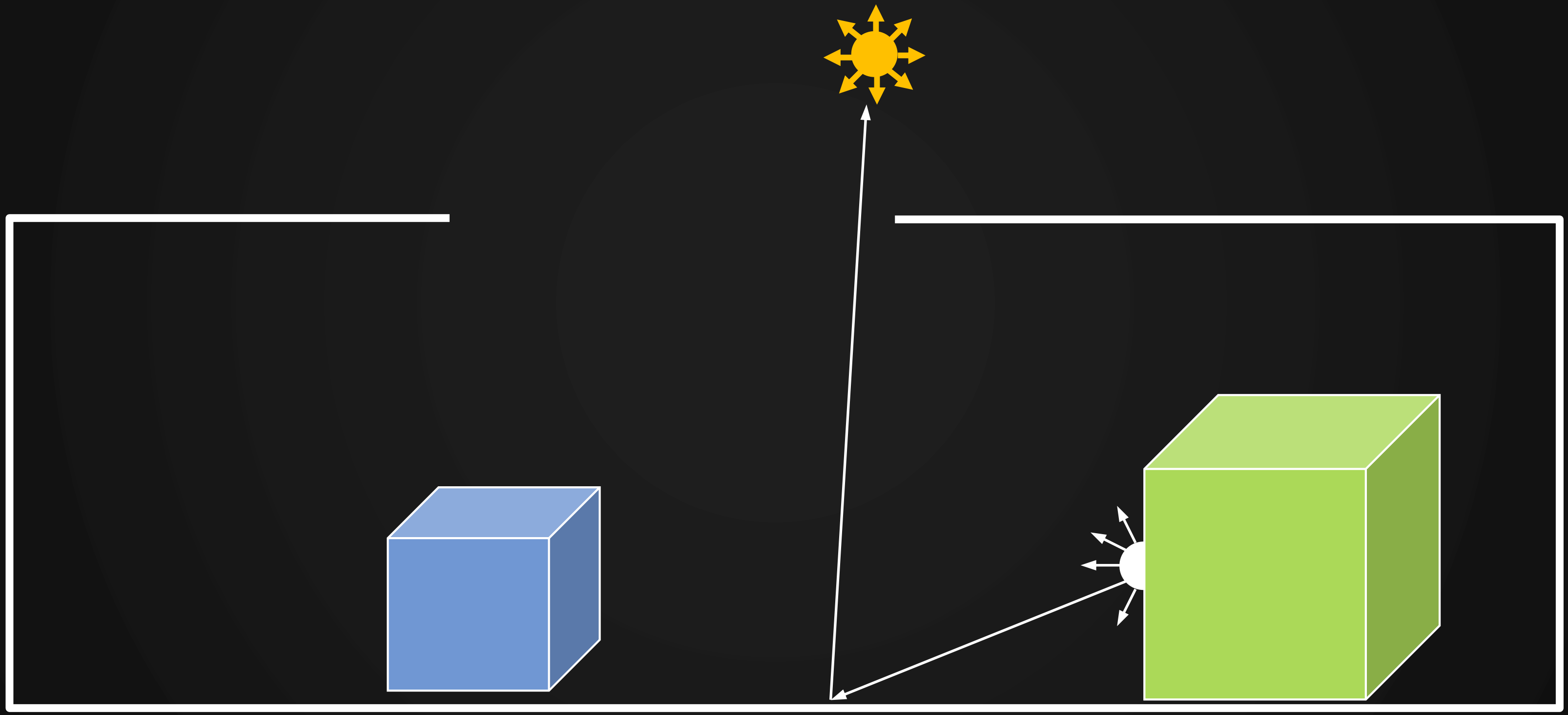
How it works

Path tracing allows the behavior of light to be simulated



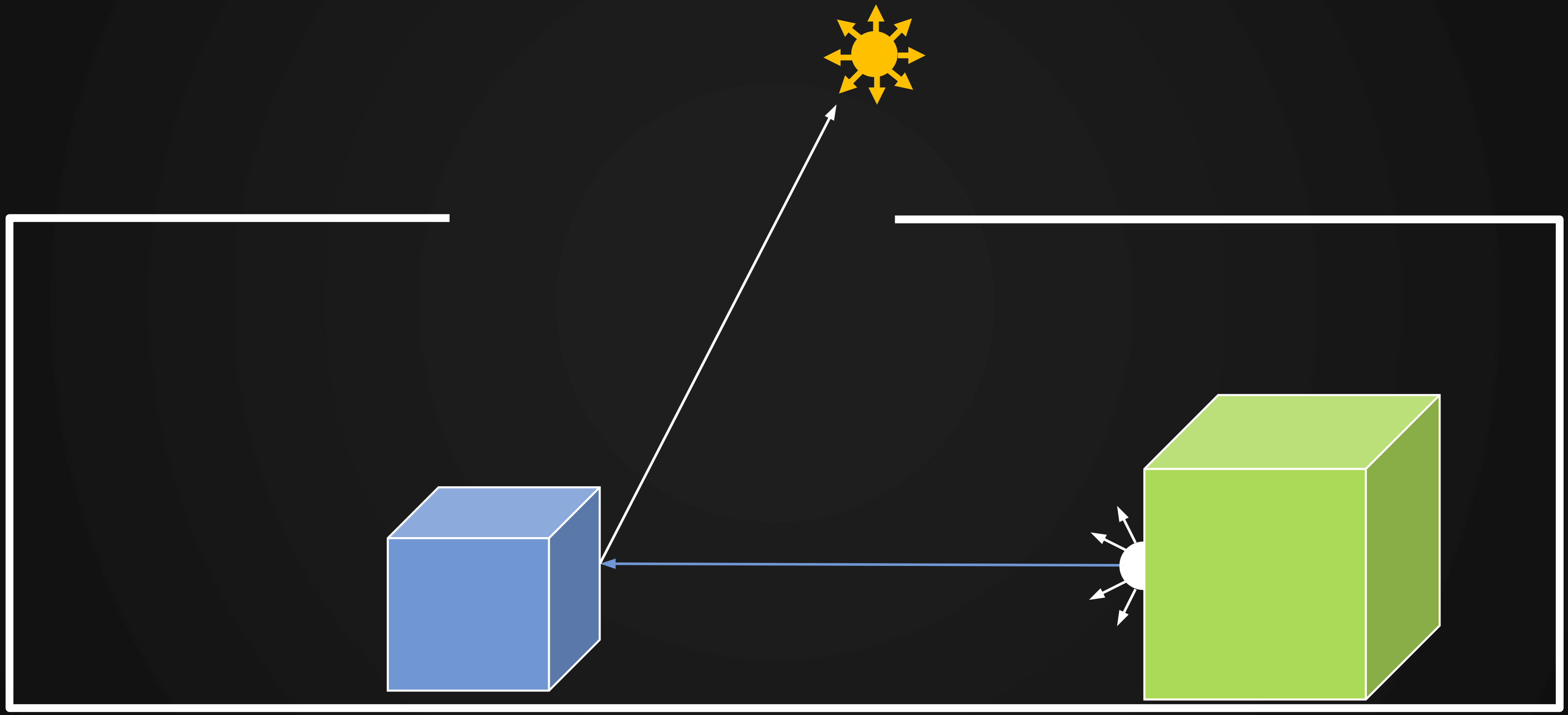
How it works

Path tracing allows the behavior of light to be simulated



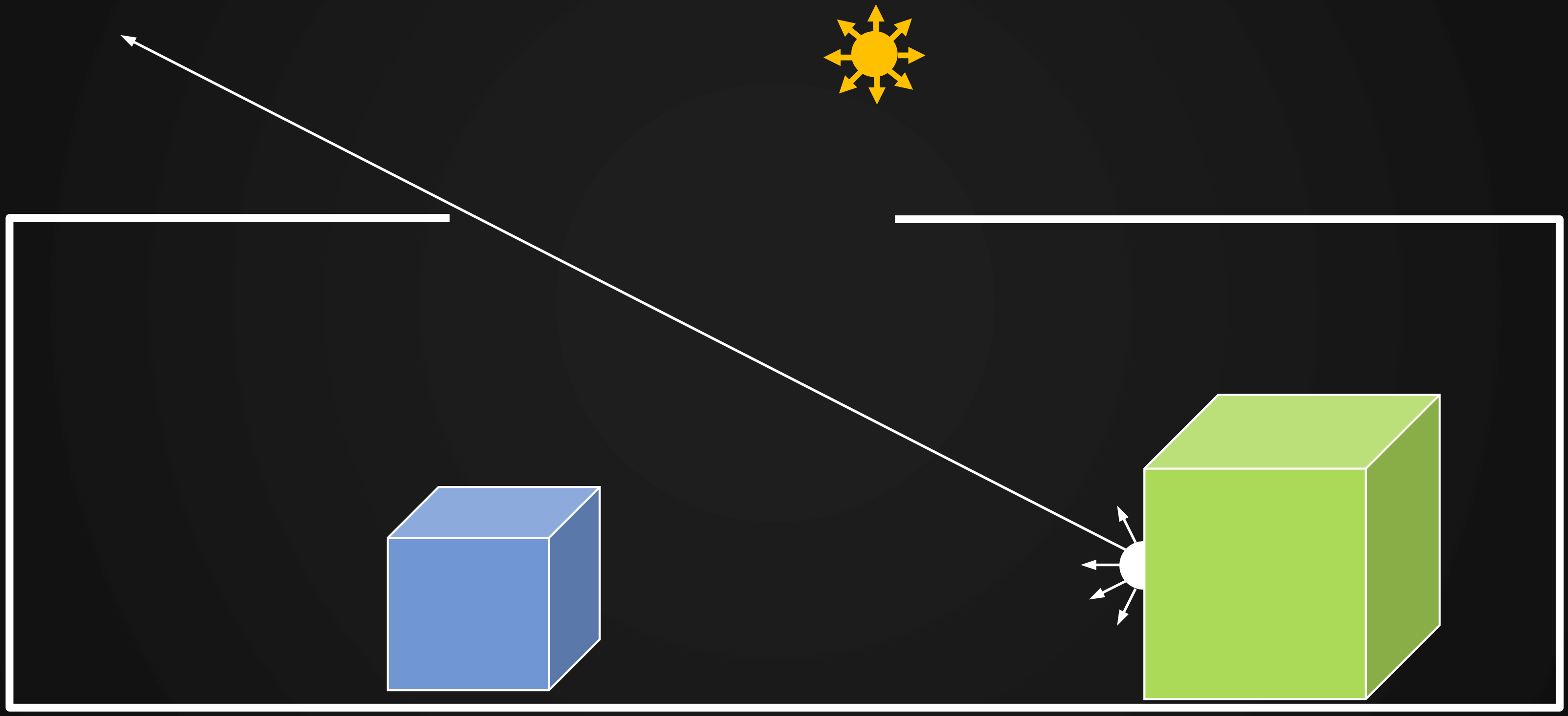
How it works

Path tracing allows the behavior of light to be simulated



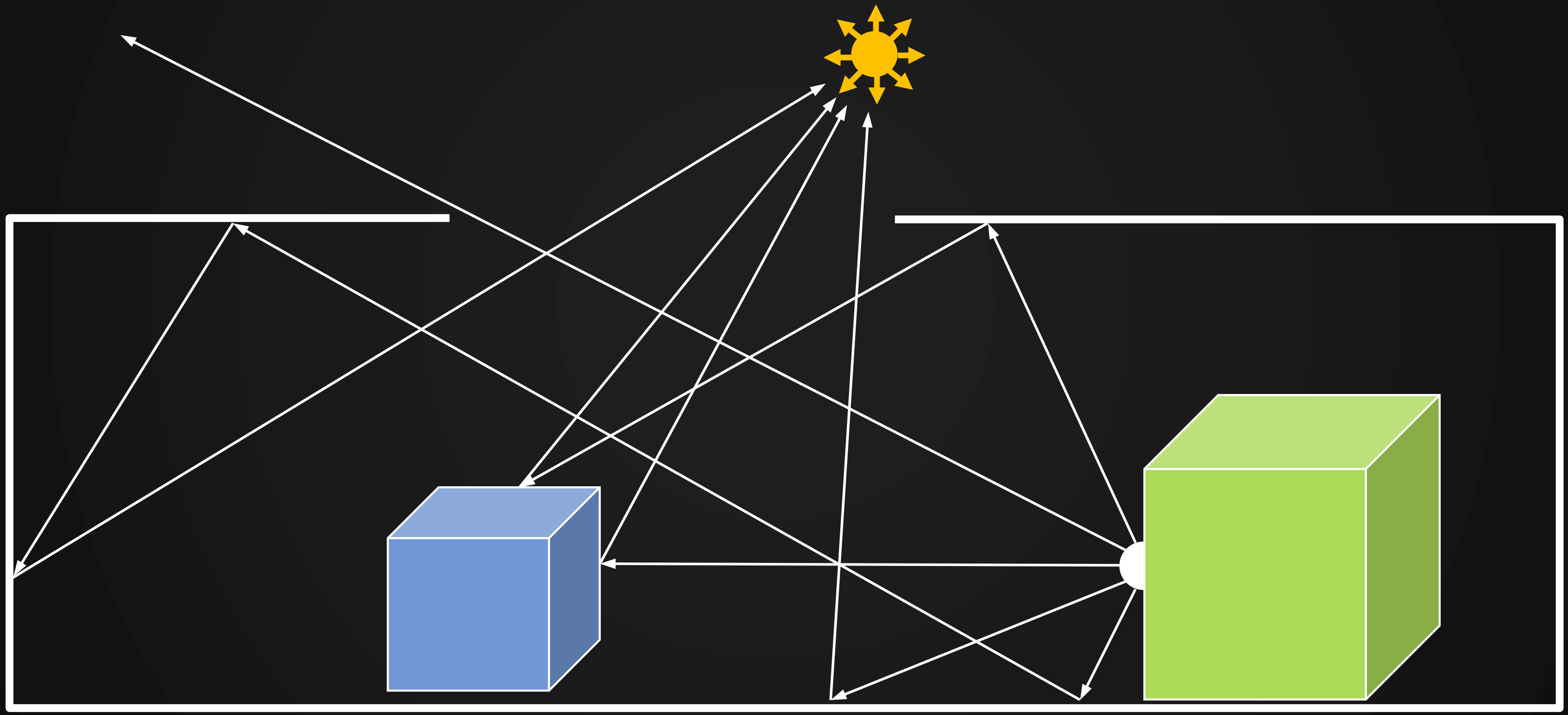
How it works

Path tracing allows the behavior of light to be simulated

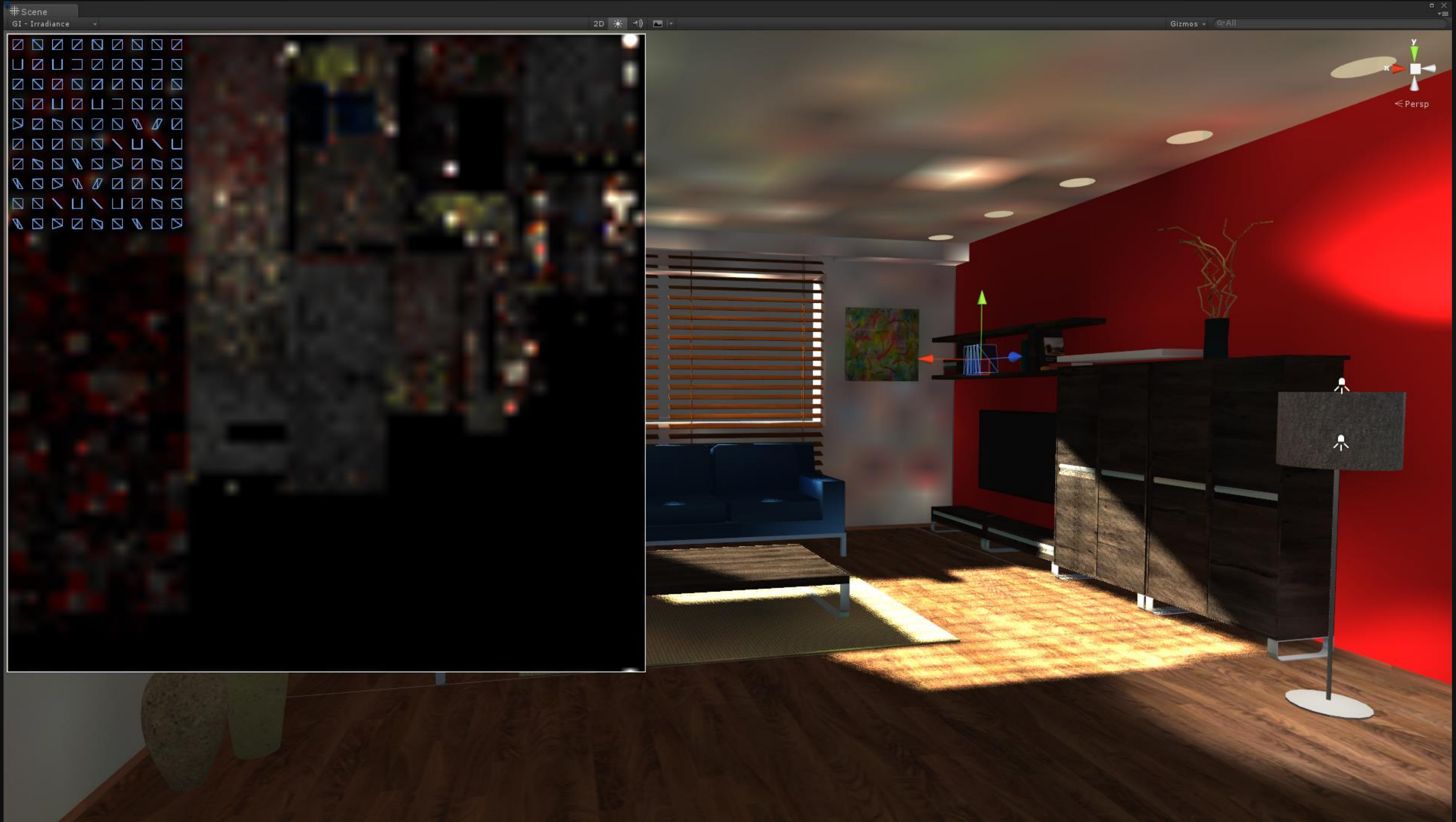


How it works

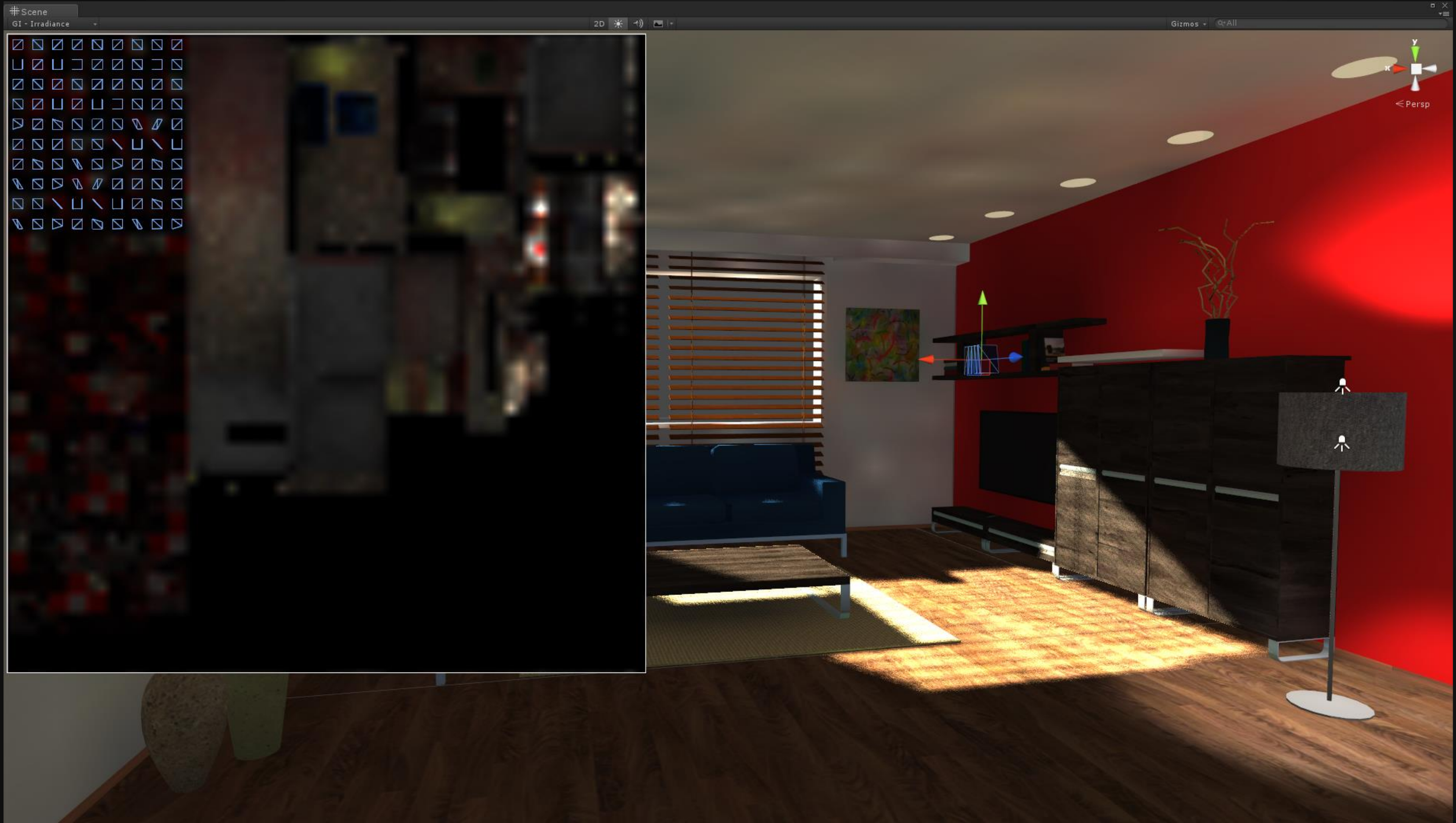
Path tracing allows the behavior of light to be simulated



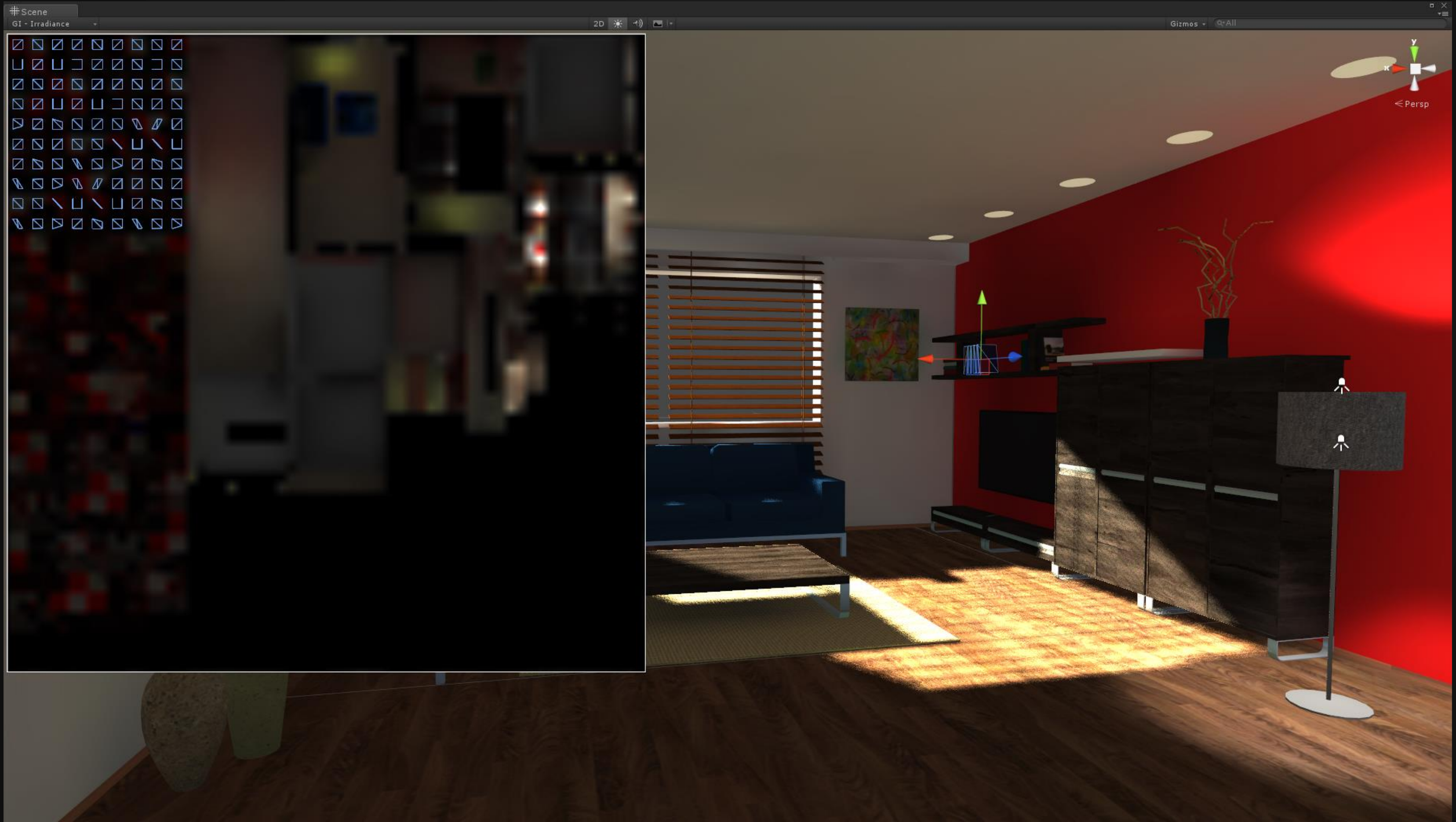
Progressive Refinement



Progressive Refinement (a moment later…)

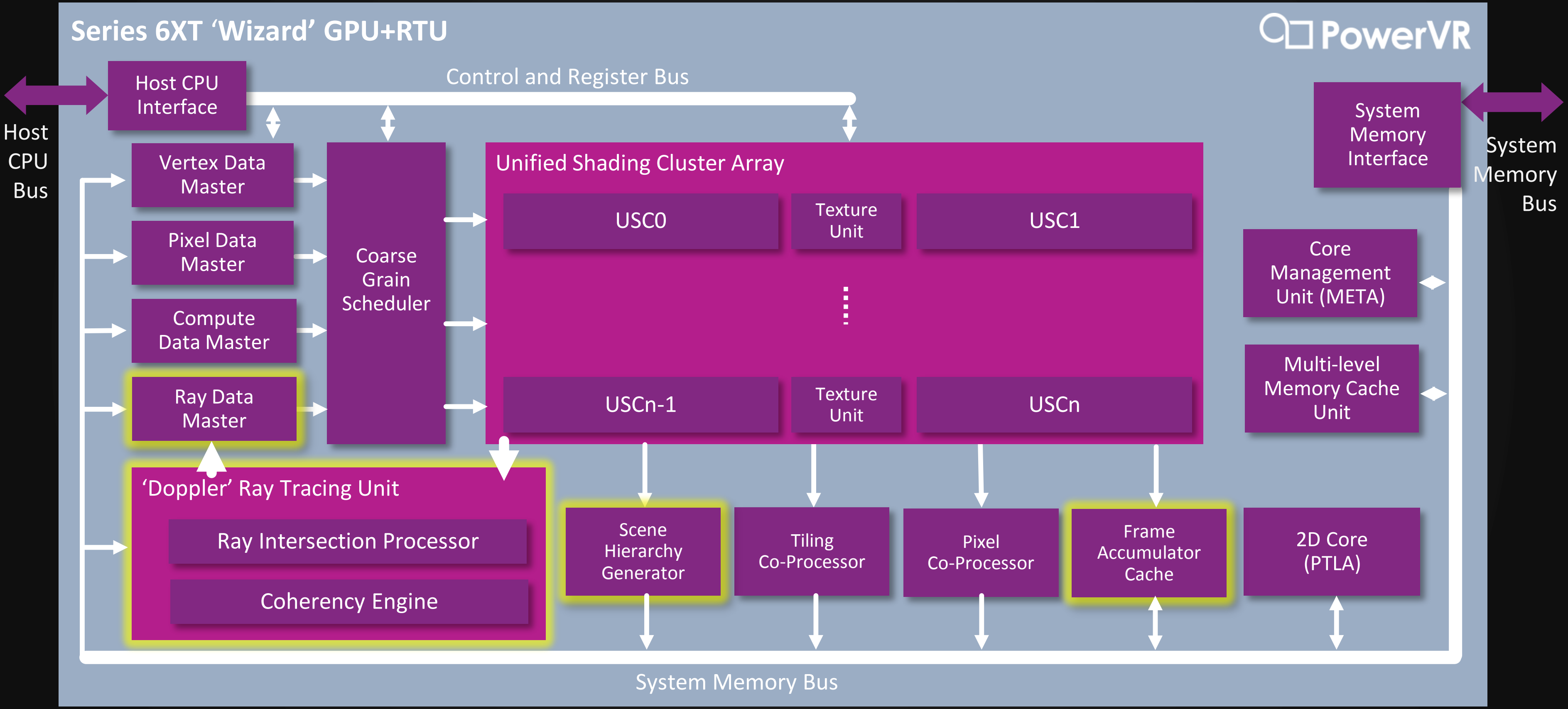


Progressive Refinement (another moment later…)



A Peek At The Future

Ray Tracing GPUs, PowerVR Wizard Family



What Ray Tracing is!

~~Myth: Ray Tracing is only for photorealistic / physically accurate rendering~~

Truth: It is just a tool. It can be used for a range of purposes

~~Myth: Ray Tracing is incompatible with rasterized graphics~~

Truth: It can be used in a rasterized game engine for certain effects

~~Myth: Ray Tracing is a less efficient way to render a given number of pixels~~

Truth: For some effects, it is computationally cheaper to ray trace

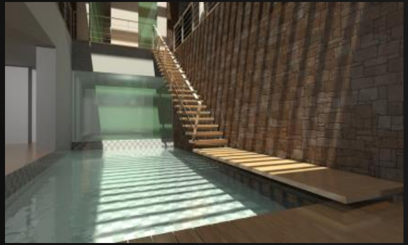
So what can you do with it?

Shadows

Reflections

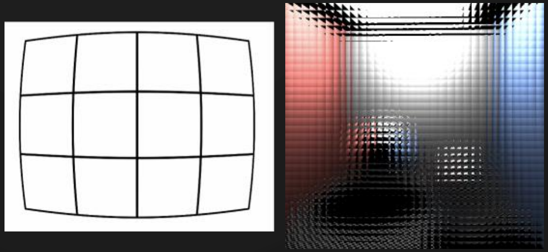
Refractions

Ambient Occlusion

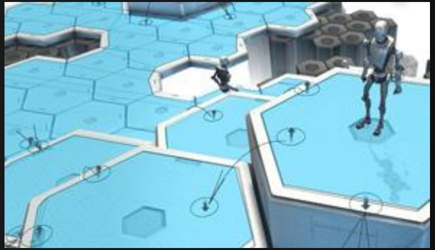


Global Illumination

Physics & Collision Detection



Virtual Reality
Lens correction, Ultra-low latency rendering, Lenticular Displays



A.I. / Line of Sight

How will ray tracing be used in games?

Many options!

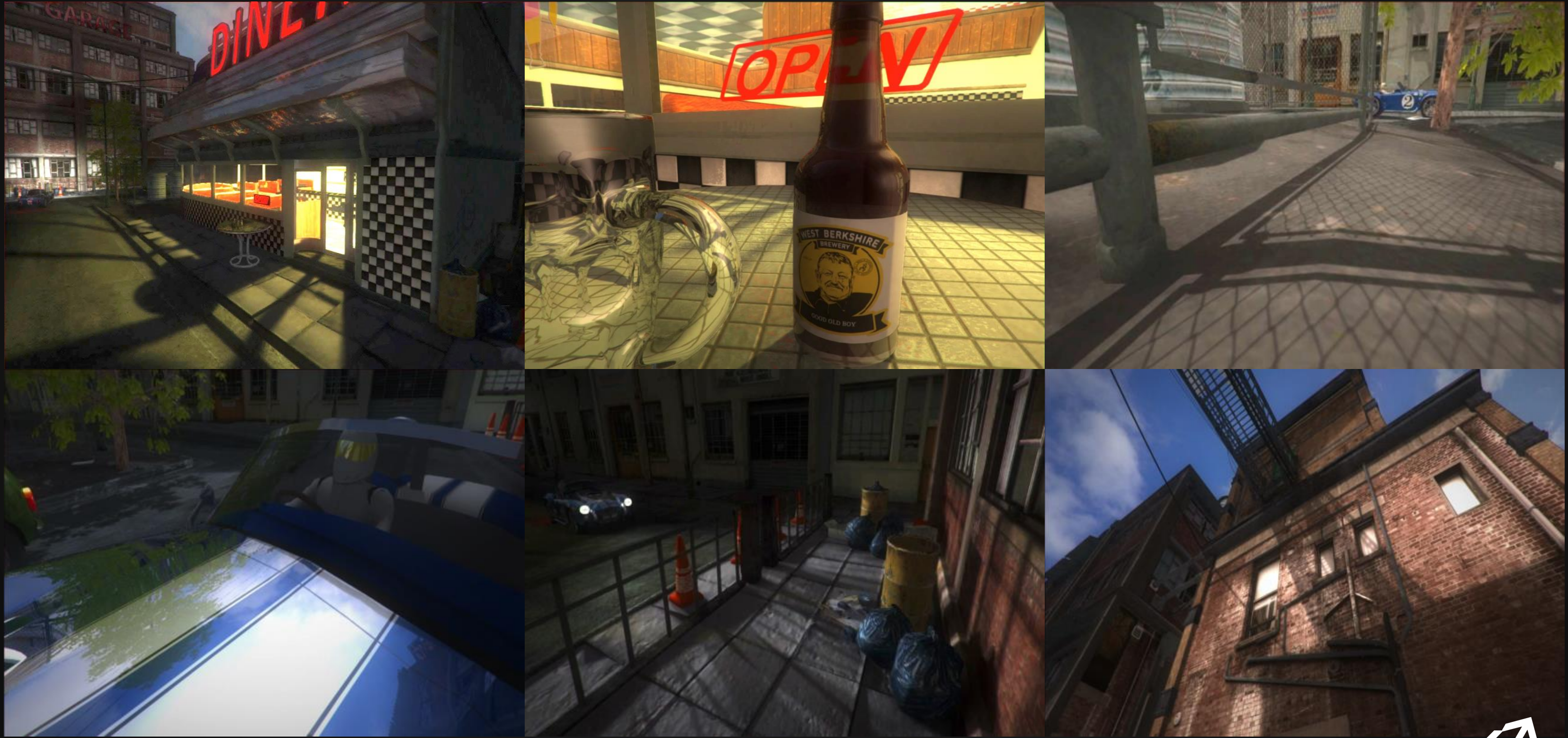
Lightmapping previews
Unity 5 Editor

Hybrid game engine

Ray trace everything,
e.g. Brigade & Arauna 2

Hybrid Rendering in Unity

Ray traced shadows and reflections in a rasterized game



Thank you

jens.fursund@imgtec.com
luke.peterson@imgtec.com

See us in the ImgTec & Unity booths

Enlighten in Unity 5.0

Sam Martin – Geomerics

Enlighten in Unity 5.0

- Practical real-time global illumination
- Enlighten at run-time
  - Dynamically light your game
  - Unique, high quality lighting
  - PC/Mac/Linux/Consoles/Mobile
- Enlighten during development
  - Fast iteration of lighting design
  - High quality lightmap baking
- Real-time, baked, and everything in between

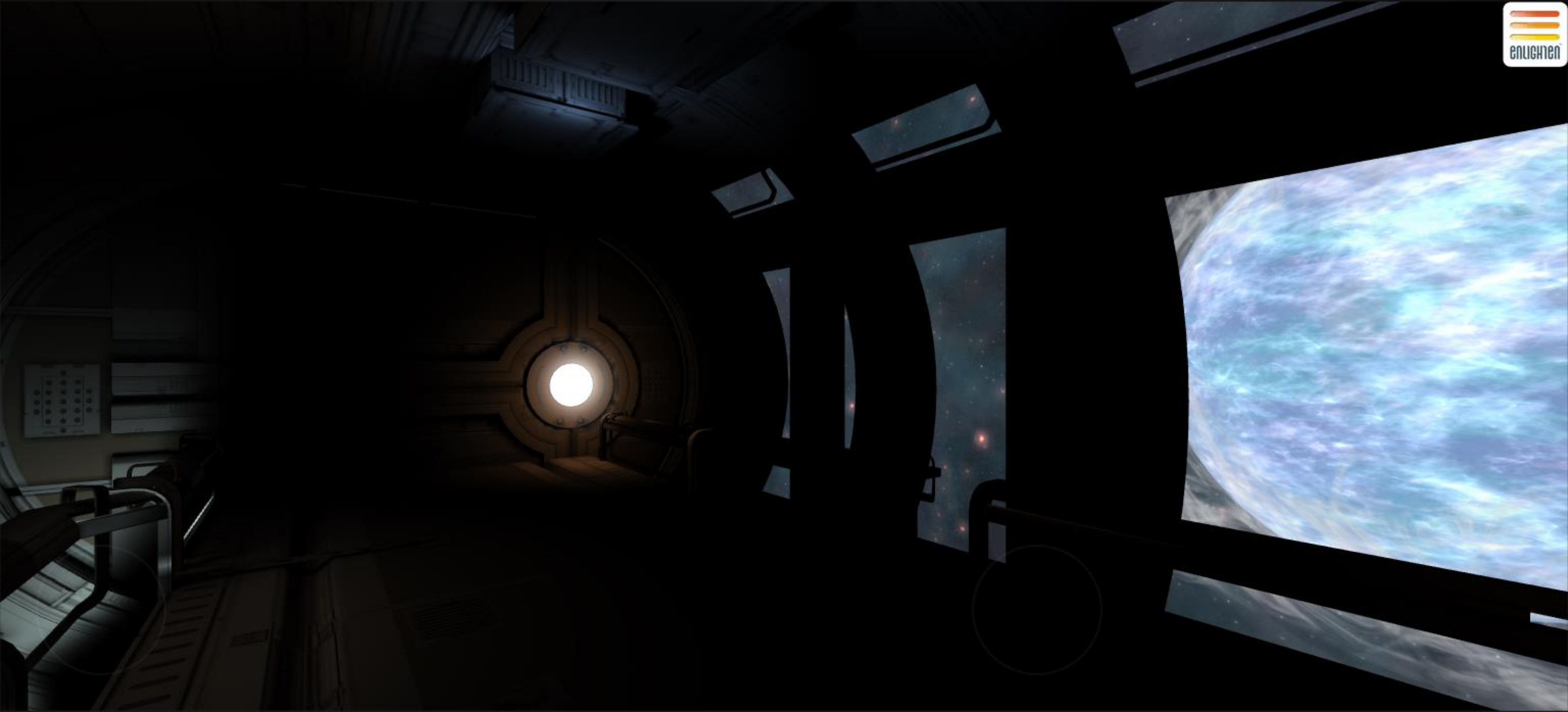
What is global illumination?

- Bounced light!
- This can come in many forms:
  - Diffuse scattering (aka radiosity); glossy reflections (specular); mirrored reflections; scattering and diffraction…
- Often multiple bounces are required
  - Particularly true indoors

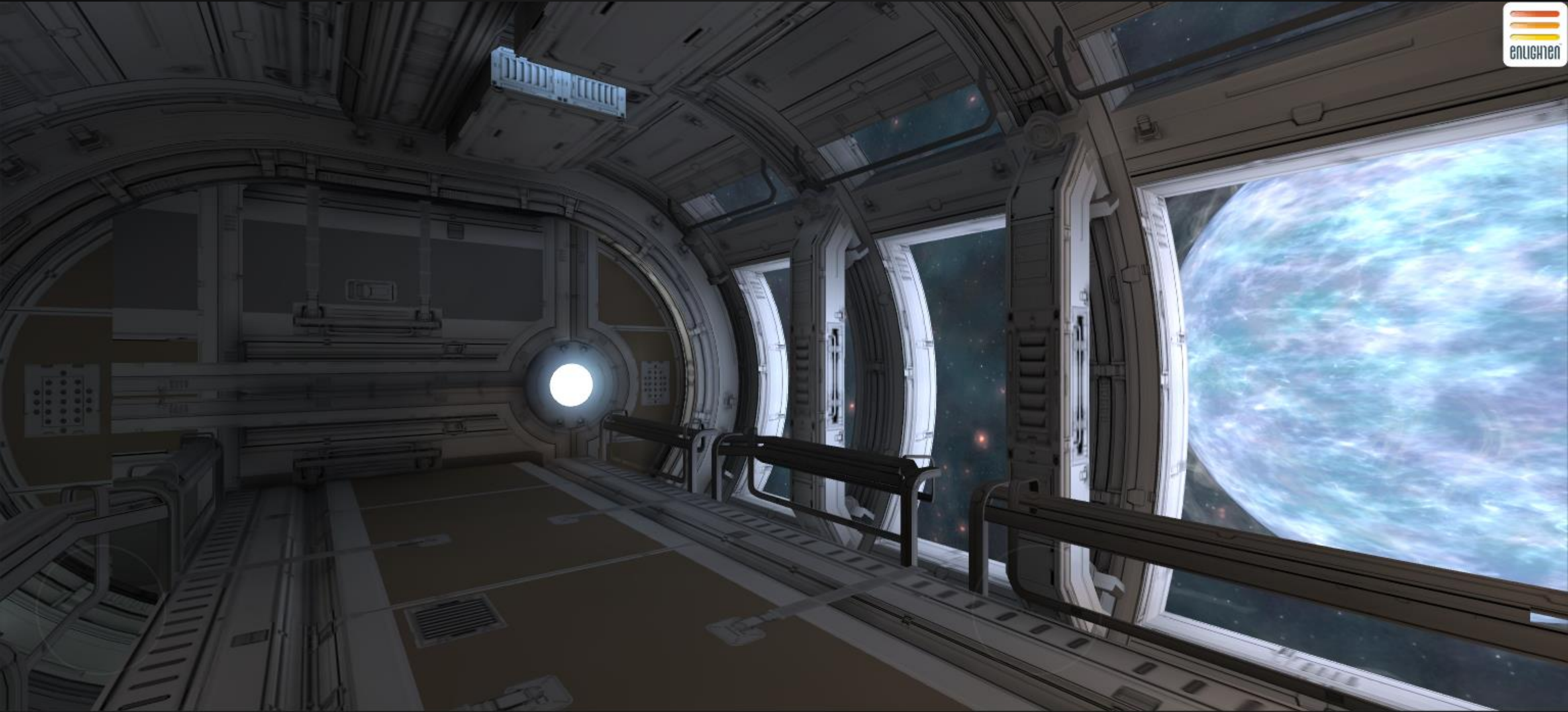
Image from Need for Speed: Rivals

Enlighten Transporter demo in Unity 5

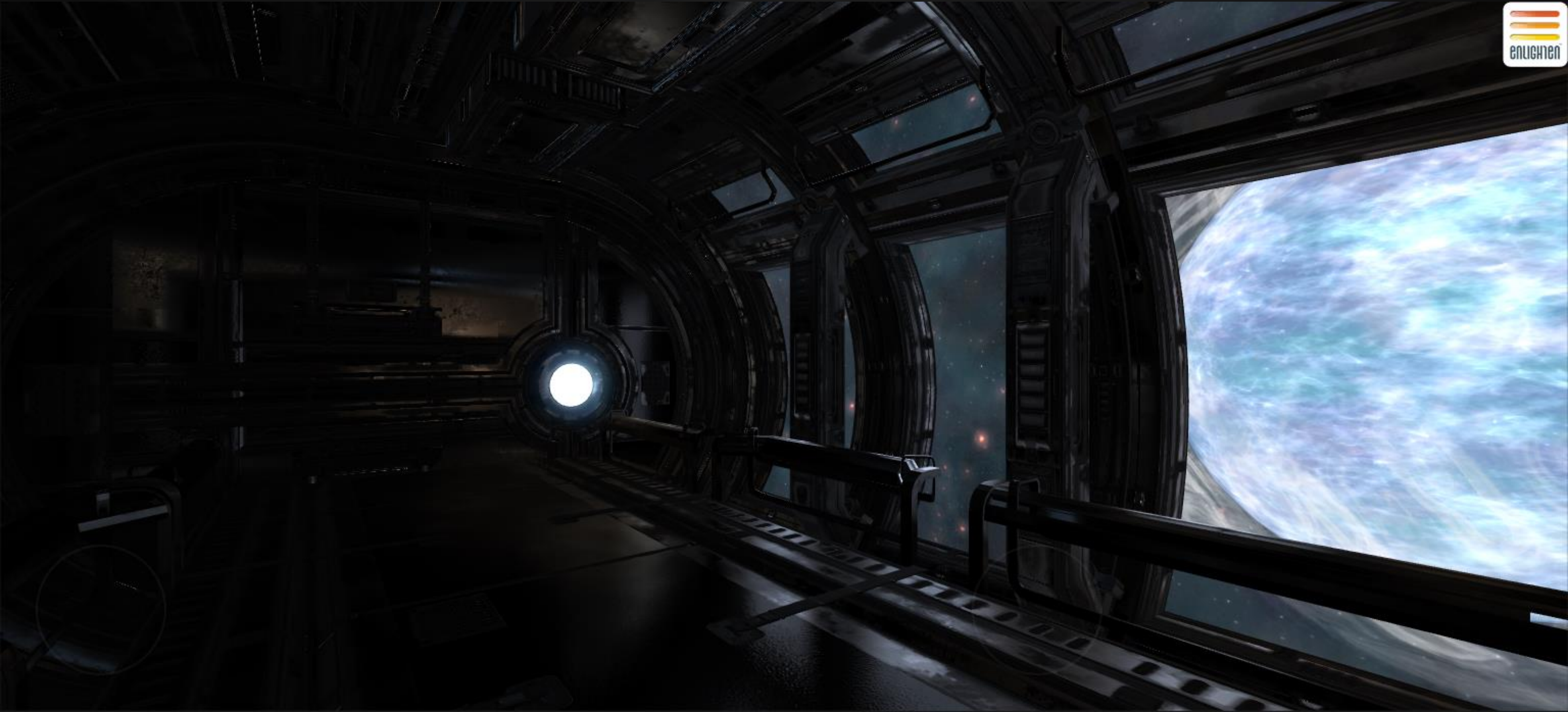
Transporter - direct only



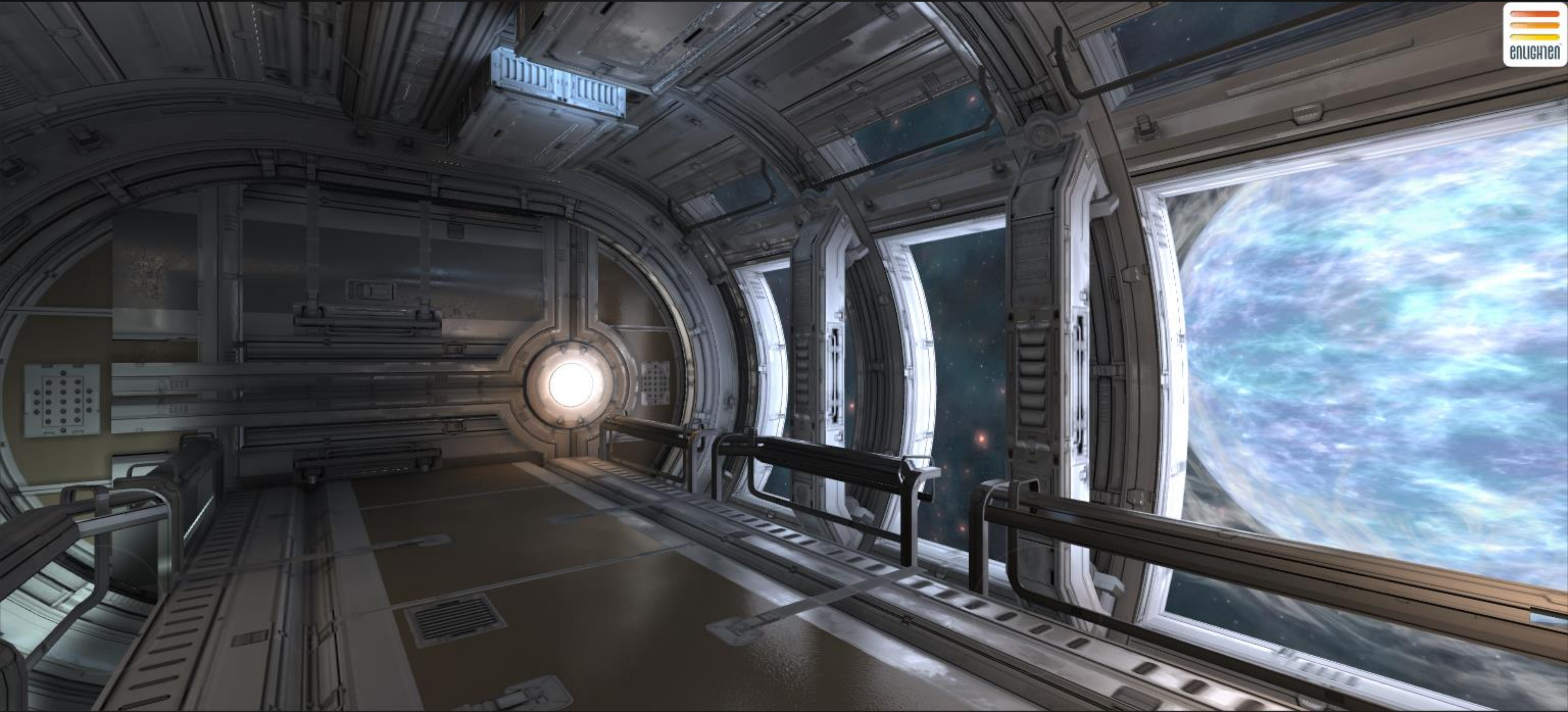
Transporter - indirect diffuse



Transporter - indirect specular



Transporter - direct + indirect



Enlighten under the hood

- Enlighten dynamically generates:
  - Lightmaps
  - Light probes (spherical harmonics)
  - Cubemaps

- Enlighten offline handles:
  - Geometry preparation
  - Ray traced direct lights and ambient occlusion
  - Interactive lightmap compositing

Enlighten principles

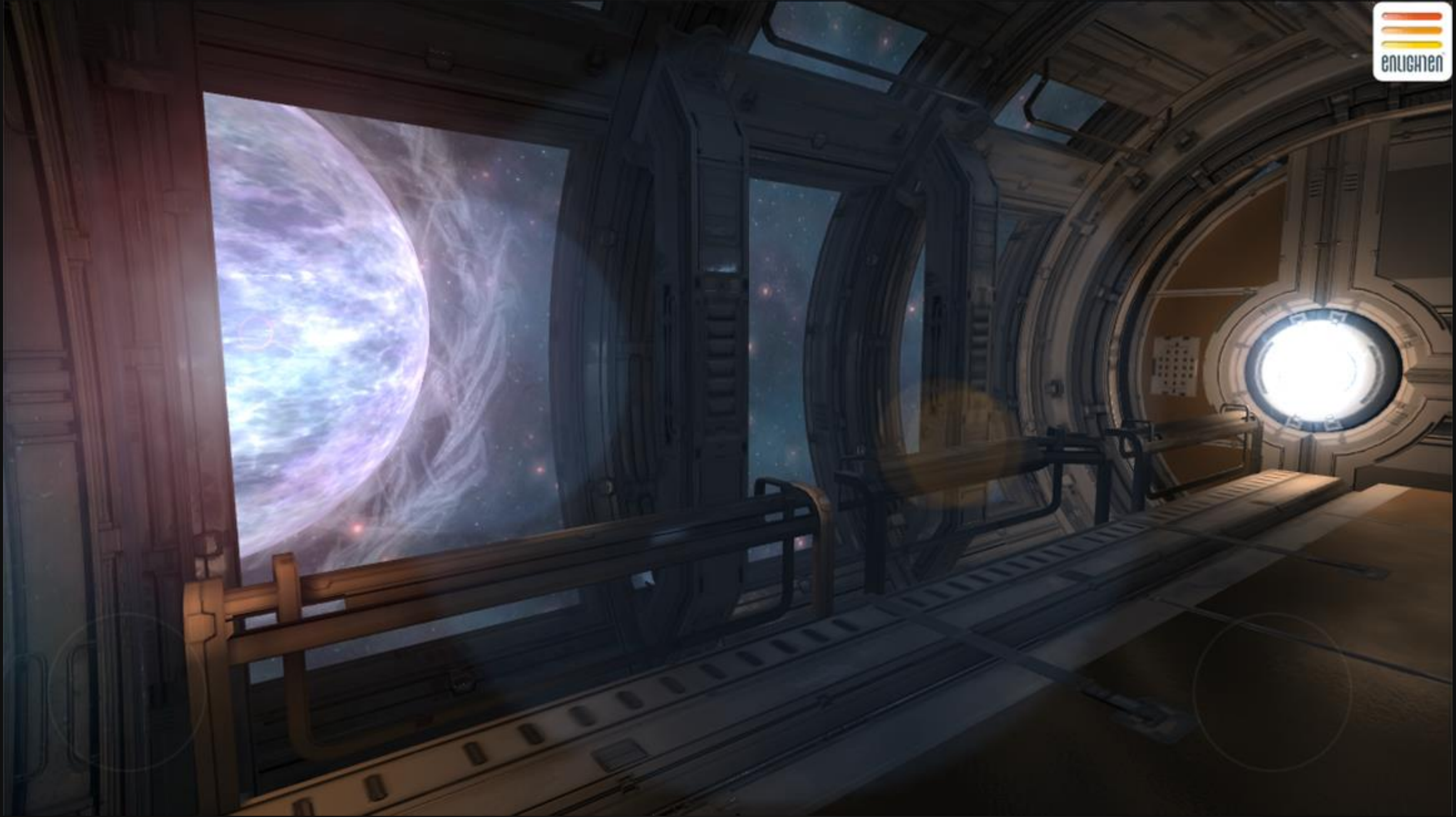
- Pay for what you change
  - No change = no cost
- Dynamic lighting must be cheap
  - Enlighten lightmaps are small and cheap to update
  - Highly optimised runtime on all platforms
  - CPU workload, no GPU overhead
- Do not sacrifice lighting quality
  - Some precomputation is better than bad lighting
- Provide artists with controls
  - Default settings → physics

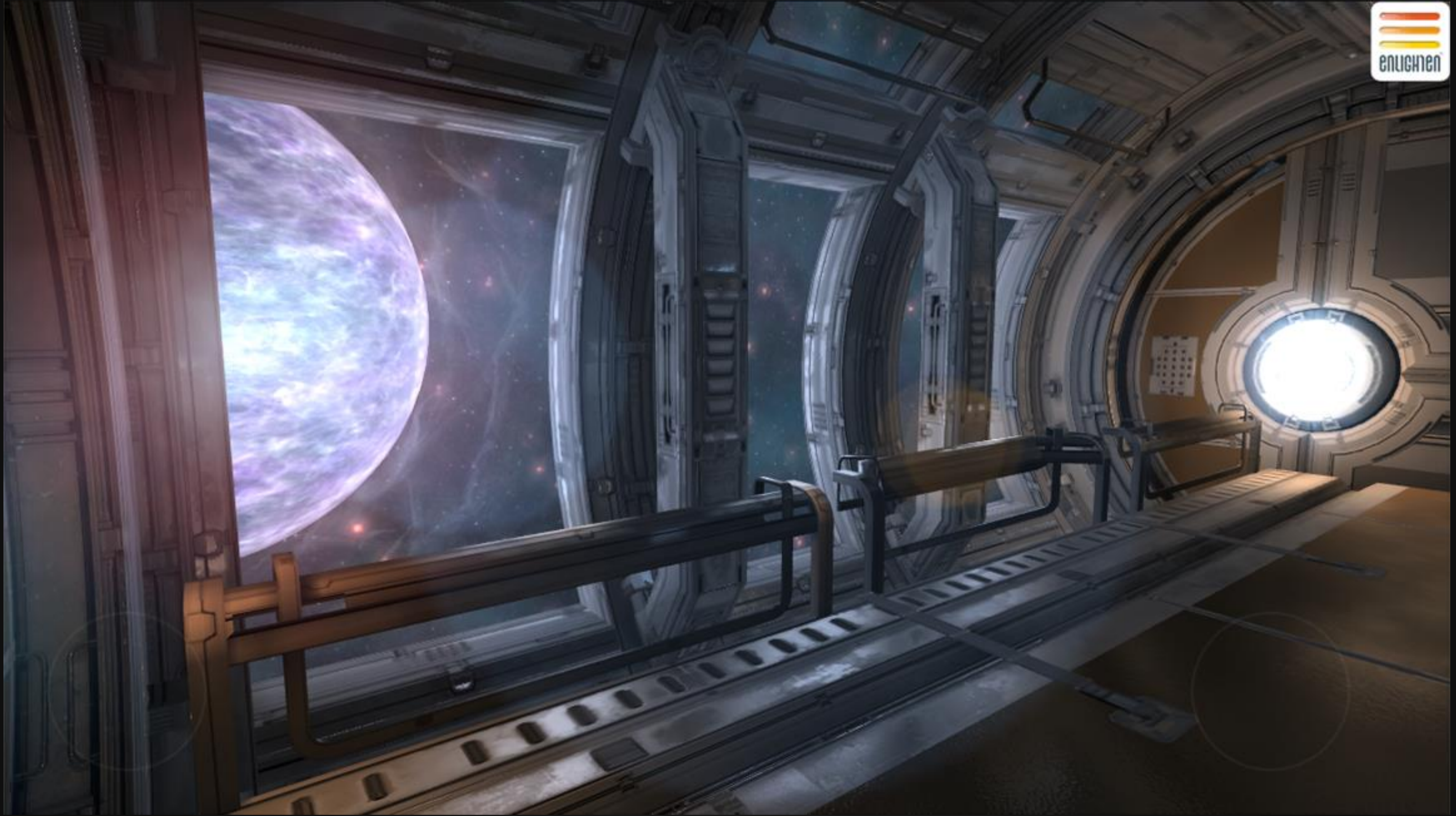
Dynamic versus static – why pick one?

- Static (baked) lightmaps:
  - + high quality soft shadows, minimal run-time cost
  - - severely limits the dynamic lighting in-game
  - - poor specular, hard edges are difficult
  - - limits iteration speed
- Dynamic lighting with Enlighten:
  - + high quality GI and specular, full artist control
  - + very flexible, fast iteration time
  - - not free (but cheap enough for mobile!)
- Enlighten covers the whole spectrum

Materials

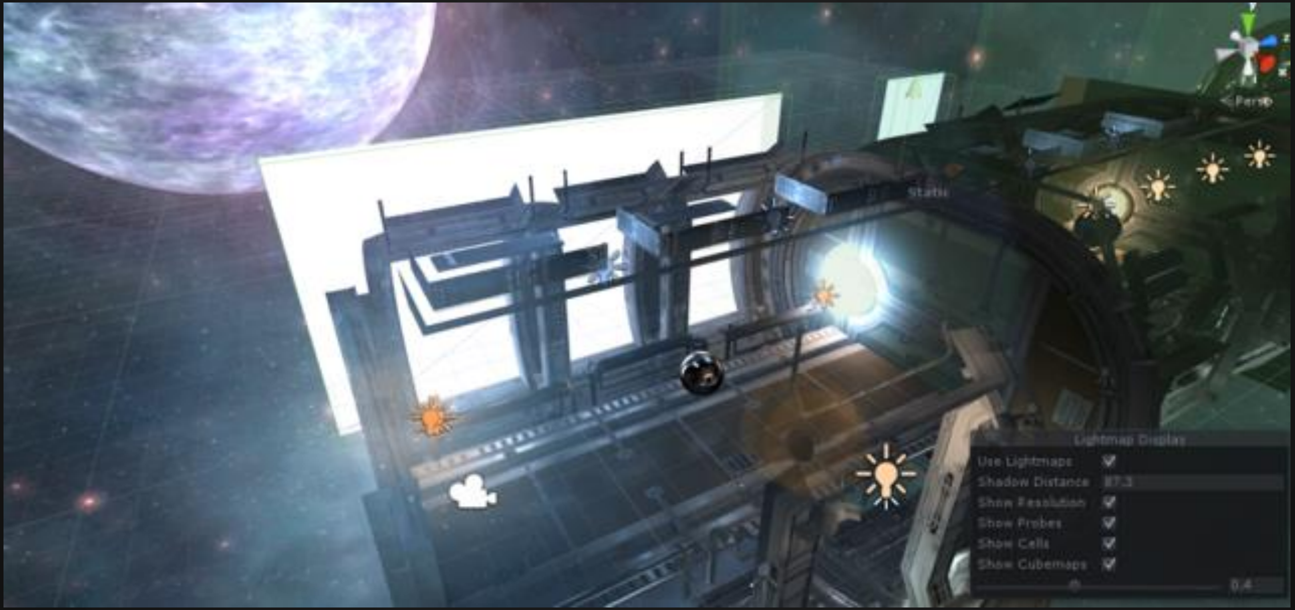
- Enlighten supports dynamic materials
- Make surfaces glow
  - Change them in-game in real-time
  - Really cheap / “free”
- Example: soft illumination in Transporter
  - Enlighten includes real-time illumination from the skybox for environment light
  - But we also had a big procedural planet courtesy of Allegorithmic…


enlighten


enlighten

Emissive surfaces

- Added hidden emissive quad
- Trivial to animate in a script



Lightmap Display

Use Lightmaps ☑

Shadow Distance 87.3

Show Resolution ☑

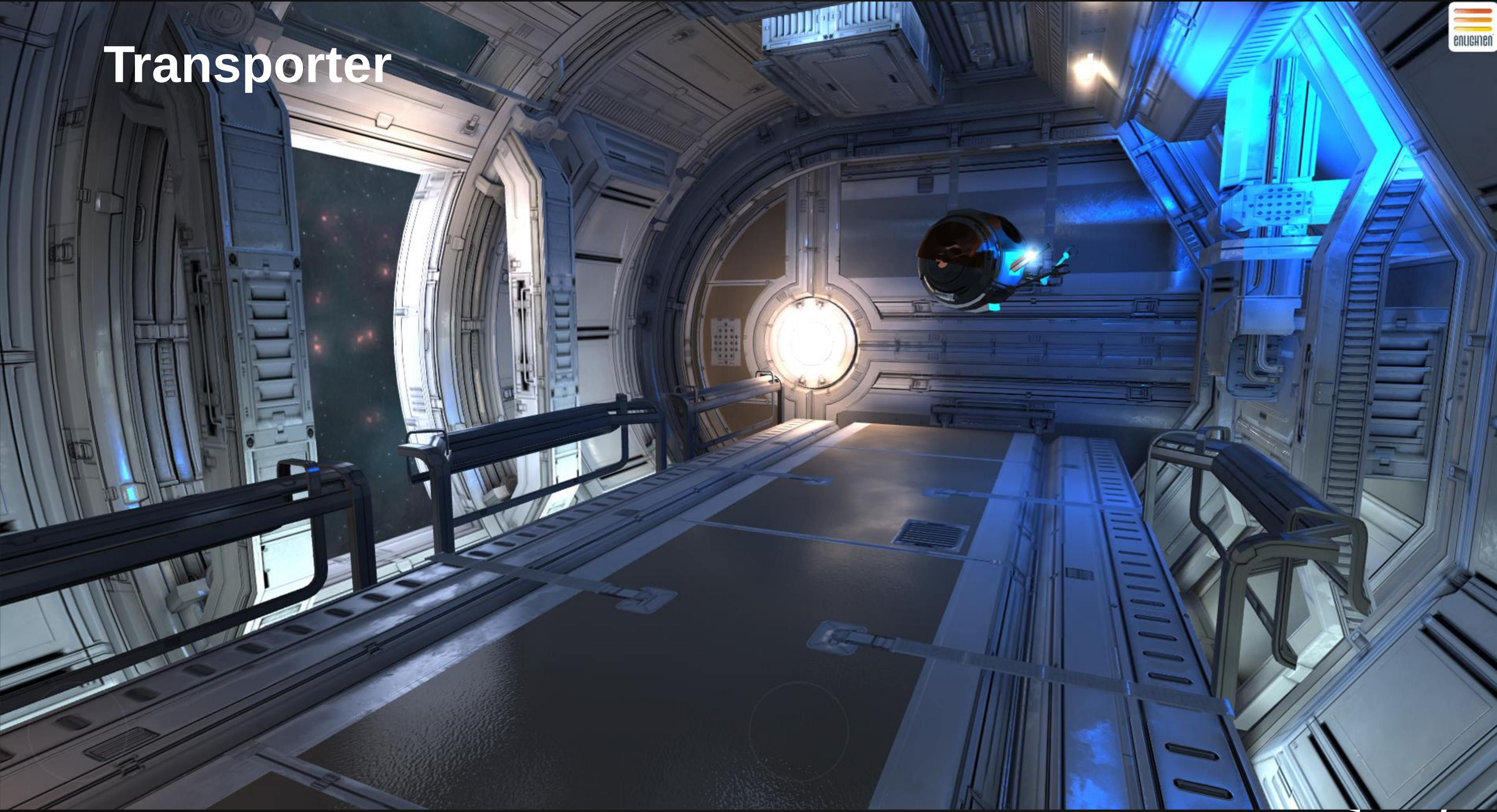
Show Probes ☑

Show Cells ☑

Show Cubemaps ☑

0.4

Transporter



DEMO

Selected titles

Eve Online

Mirror’s Edge 2

The Future

- Start of a long-term relationship between Unity and Geomerics
- Geomerics ongoing R&D
  - Efficient real-time shadowing on mobile and next-gen consoles
  - High performance mobile rendering
  - Even more detailed lighting
- Enlighten continues to improve

- Come see: “The Revolution in Mobile Game Graphics”
  - Wednesday, 11.00am, Room 3014 West Hall

Thank you

sam.martin@geomerics.com
chris.doran@geomerics.com

See us in BS 2622, North Hall
and in the ARM & Unity booths

Static Lightmap Baking Improvements in Unity 5.0

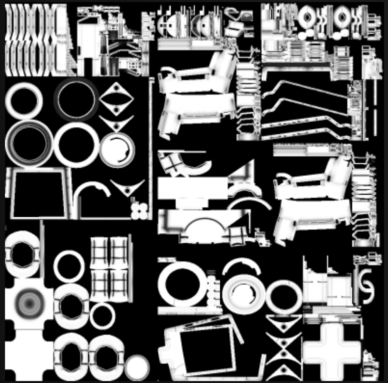
- If you don’t want to use all the new fancy stuff, there’s still baking…
- Now powered by Enlighten
- Feature parity with baking in 4.x
- Lightmaps can seamlessly be mixed with dynamic lightmaps
- In many cases bakes faster
- Tighter integration - supported in the iterative workflow

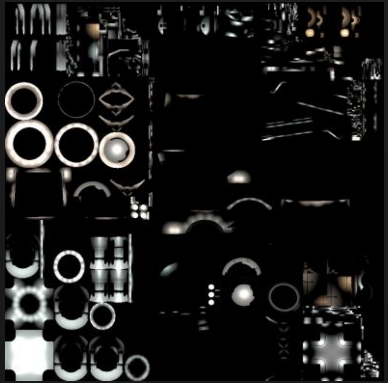
Static Lightmap Baking Improvements in Unity 5.0

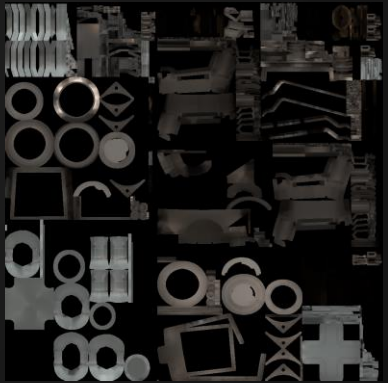
- No more “export scene → bake → import lightmaps”
- Baking is integrated more tightly
- Changes to the scene are often integrated more quickly, e.g.
    - modifying a baked light updates only *that* light – no full rebake
    - editing materials only updates indirect lighting
    - compositing settings can be interactively modified
- Basically minimal bakes are guaranteed
- But is conservative

Static Lightmap Baking Improvements in Unity 5.0

- Baking now produces lightmaps in components
  - ambient occlusion
  - direct lighting
  - indirect lighting

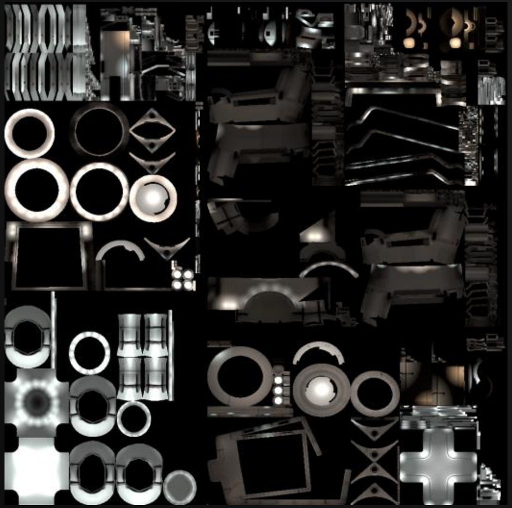


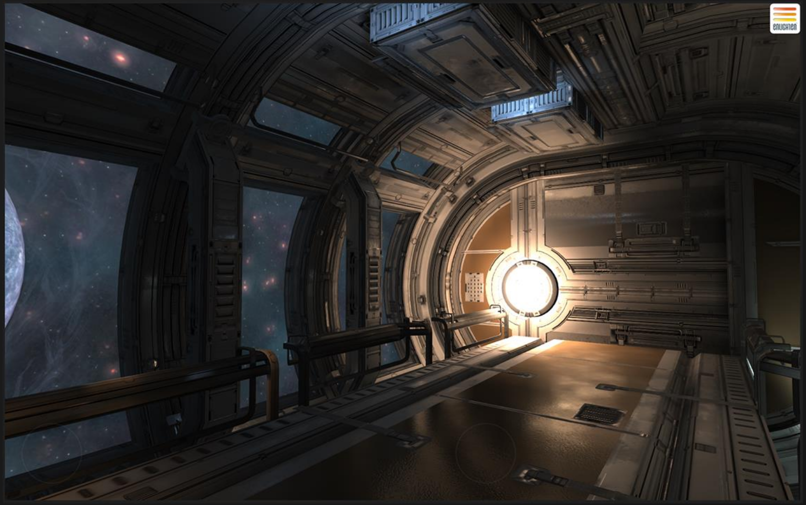




Static Lightmap Baking Improvements in Unity 5.0

- These are composited into a single texture
- Output = D_{scale} * Direct + I_{scale} * Indirect * AO^{aoExp}
- Adjust compositing parameters in the Editor interactively





Integration with Standard Shader in 5.0

- It works...
- shader supports:
  - baked lightmaps
  - baked light probes
  - reflection probes
  - dynamic lightmaps NEW
  - dynamic light probes NEW
  - dynamic cubemaps NEW

The future…

- Keep improving the iterative workflow
- Cloud computation?
- More dynamic features (area lights)
- Layered lightmaps + in-game compositing
- Go make some awesome games!

Thank you for listening!

?