Unity 5
Graphics Smörgåsbord

Aras Pranckevičius
Rendering Plumber

unity

Overview

- Unity 5 adds a lot of graphics goodies!
- Let’s run through them real fast

Physically Based Shading

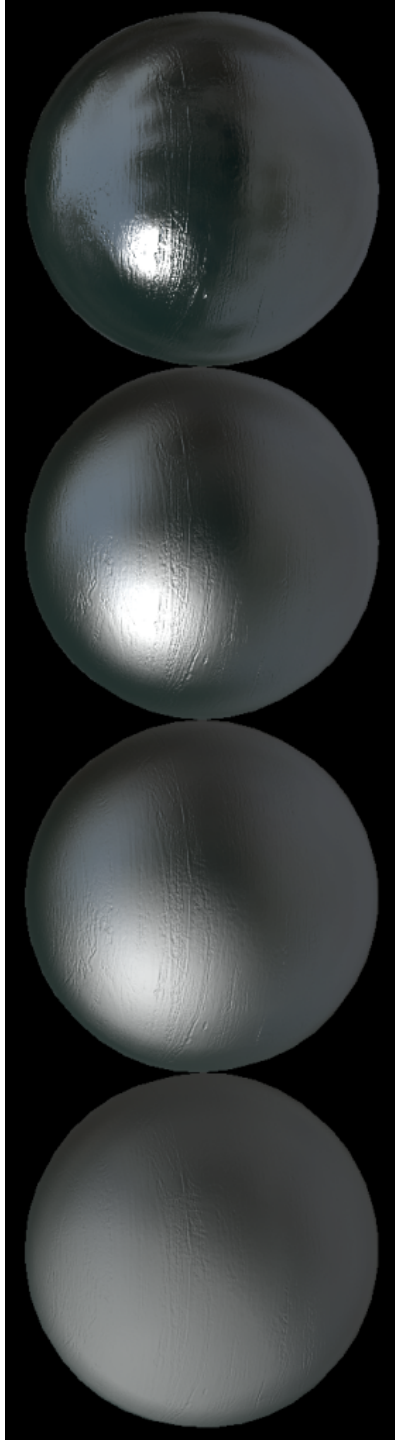
unity

Motivation for PBS

- Predictable look under different lighting conditions
- Less guesswork, fewer ad-hoc parameters
- More interop with other tools (Toolbag, Substance, …)
- Allow capture/scanning of real world data.

PBS: Energy Conservation

- Don’t reflect more light than you receive
- Sharper reflections = stronger; blurrier reflections = dimmer
- Specular takes away from diffuse



PBS: Specular & Fresnel

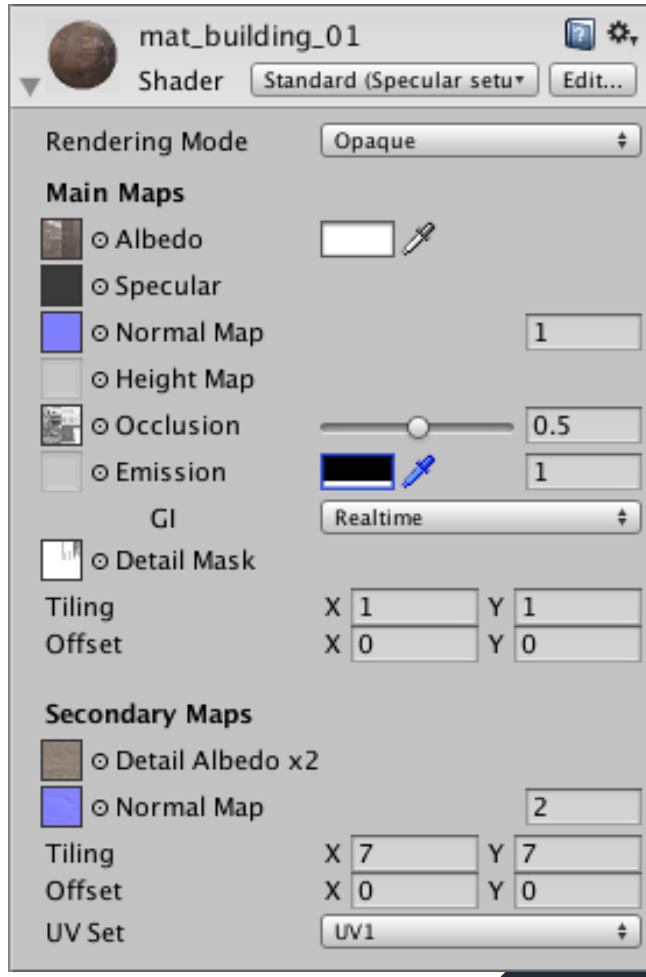
- Everything has specular
- Everything has Fresnel
  - “Surface becomes more reflective at grazing angles”

PBS: Image Based Lighting

- Not *strictly* a requirement for PBS but…
- It's good for approximating the "entire lighting environment"

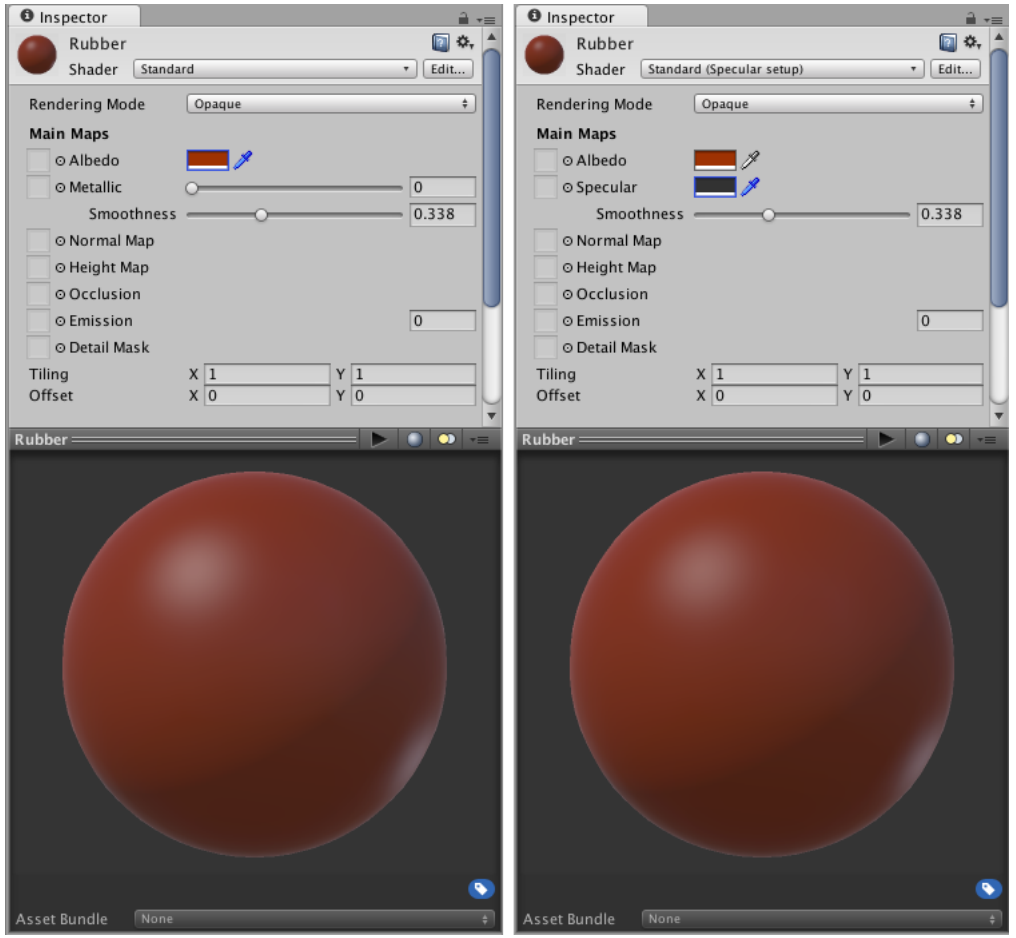
Standard Shader

- Built-in shader for most everyday materials
- No more hunting for “which shader to use?”
- Features driven by streamlined inspector



Metallic vs Specular

- Metallic:
  - Albedo (RGB)
  - Metallic (R)
    Smoothness (A)
- 0 metallic:
  - Diffuse=albedo
  - Specular 4% gray
- 1 metallic:
  - Diffuse=black
  - Specular=albedo

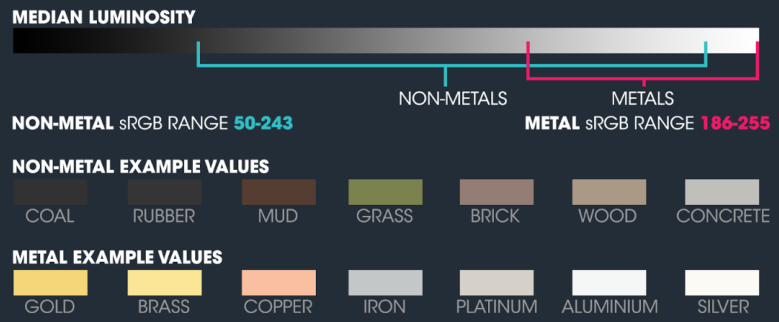


- Specular:
  - Diffuse (RGB)
  - Specular (RGB)
    Smoothness (A)

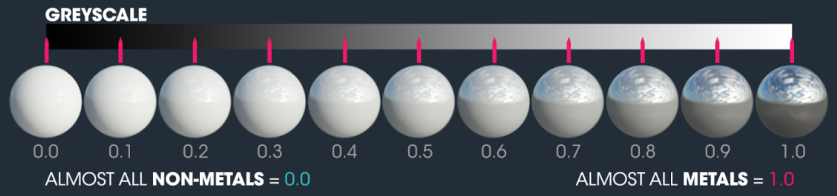
unity

SHADER CALIBRATION SCENE
METALLIC VALUE CHARTS

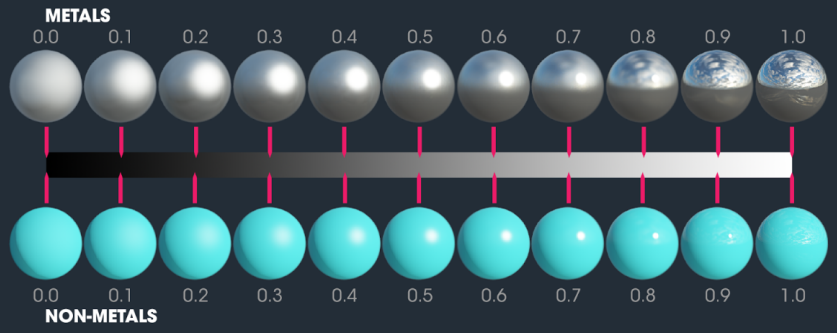
ALBEDO RGB
ALBEDO DEFINES THE **OVERALL COLOUR** OF AN OBJECT
VALUES USUALLY MATCH THE PERCEIVED COLOUR OF AN OBJECT



METALLIC R
METALLIC DEFINES WHETHER A SURFACE APPEARS TO BE **METAL** OR **NON-METAL**
WHILST PURE SURFACES WILL BE EITHER 0.0 **OR** 1.0, BEAR IN MIND FEW PURE, CLEAN, UNWEATHERED MATERIALS EXIST IN REAL LIFE
WHEN **TEXTURING** A METALLIC MAP, THIS VALUE WILL ALWAYS BE **GREYSCALE** AND IS STORED IN THE **R CHANNEL** OF AN RGB FILE



SMOOTHNESS A
SMOOTHNESS DEFINES THE PERCEIVED **GLOSSINESS** OR **ROUGHNESS** OF A SURFACE
FOR TEXTURES, THIS IS STORED AS THE ALPHA CHANNEL OF THE **METALLIC MAP**

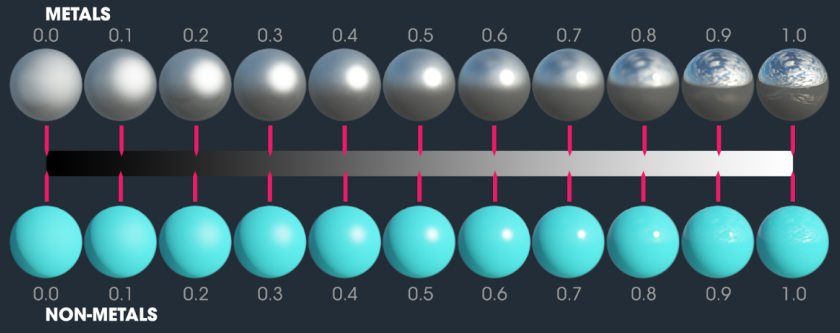


SHADER CALIBRATION SCENE
SPECULAR VALUE CHARTS

ALBEDO RGB
ALBEDO DEFINES THE **DIFFUSE REFLECTIVITY** OF A SURFACE
FOR **NON-METALS** THIS WILL USUALLY BE THE PERCEIVED COLOUR OF THE OBJECT. FOR **METALS** THIS IS USUALLY BLACK

SPECULAR RGB
SPECULAR DEFINES THE **SPECULAR REFLECTIVITY** OF A SURFACE
NON-METALS ARE USUALLY A DARK GREY OF ~55 in sRGB. FOR **METALS**, THIS IS USUALLY THE PERCEIVED COLOUR OF THE METAL

SMOOTHNESS A
SMOOTHNESS DEFINES THE PERCEIVED **GLOSSINESS** OR **ROUGHNESS** OF A SURFACE
FOR TEXTURES, THIS IS STORED AS THE ALPHA CHANNEL OF THE **SPECULAR MAP**

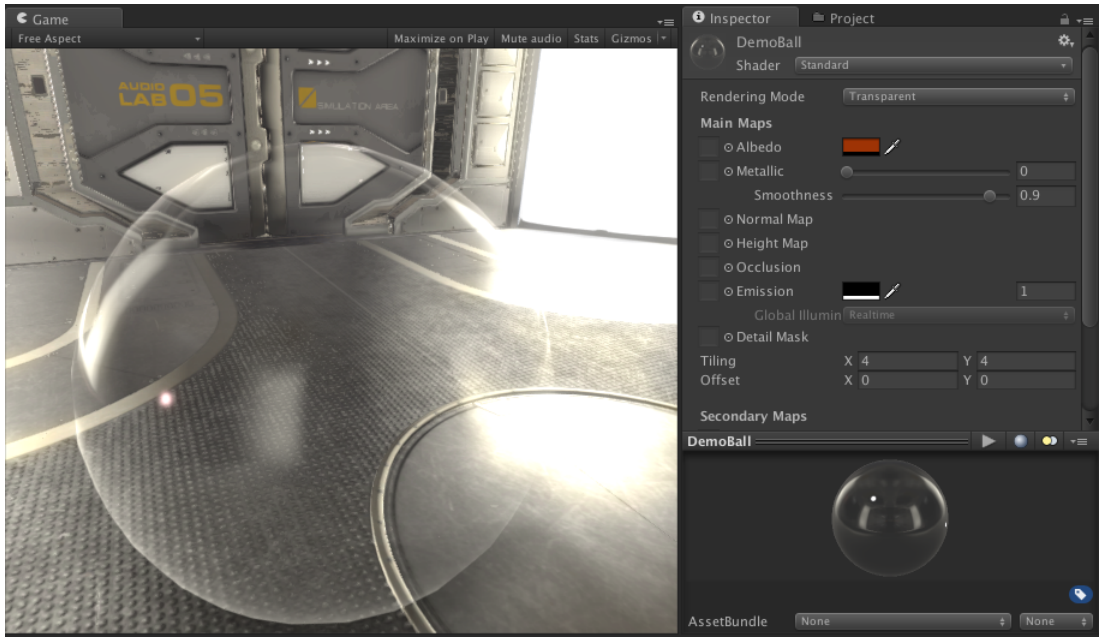


Transparency

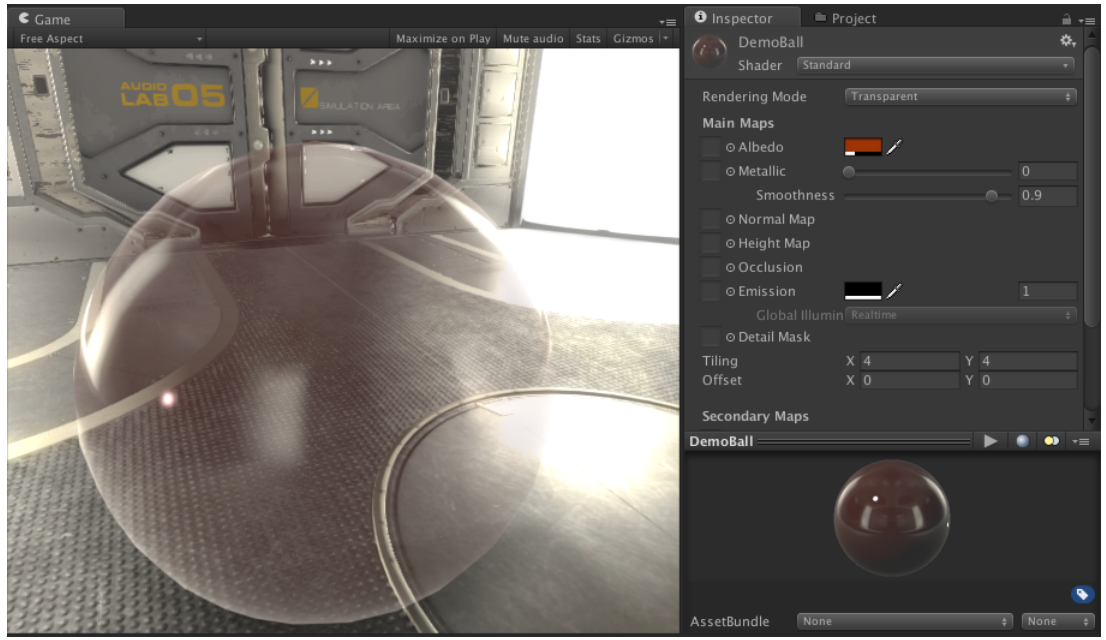
- Physically plausible “Transparent” mode
  - Glass, water, …
  - Specular & reflections visible, even at alpha=0!
- Gameplay/Effects oriented “Fade” mode
  - For fading out objects
  - Invisible at alpha=0

Transparency: Transparent

Alpha=0



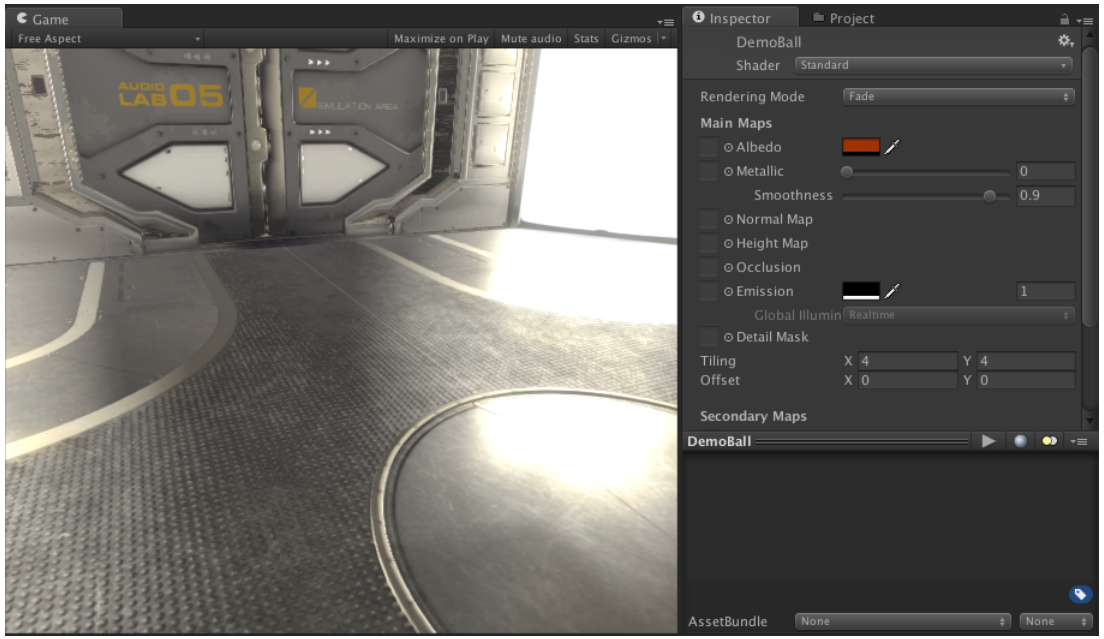
Alpha=0.3



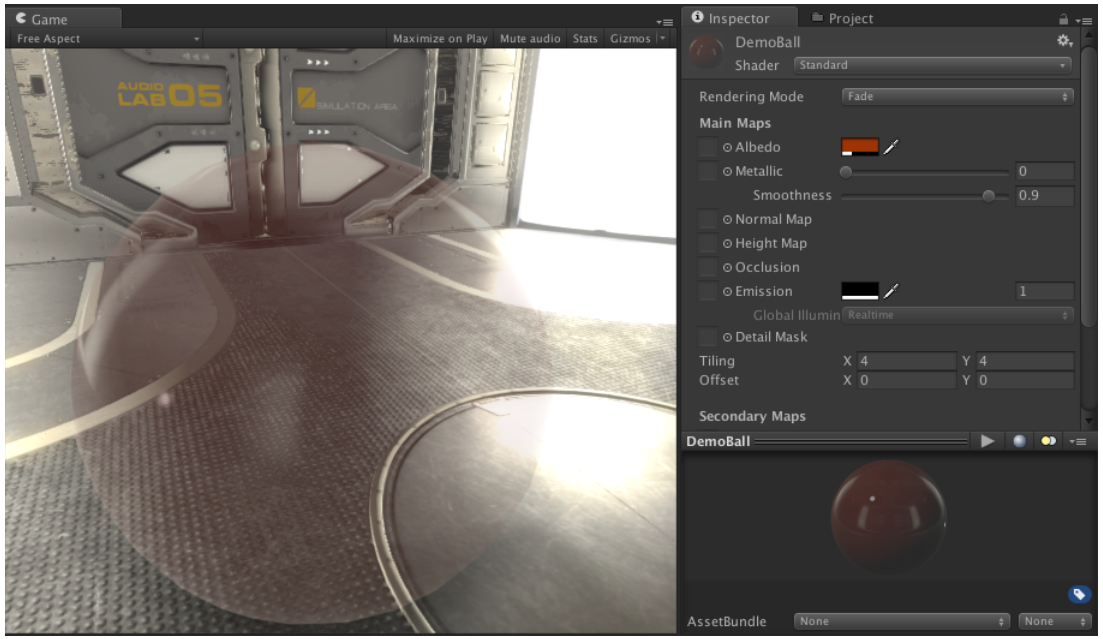
unity

Transparency: Fade

Alpha=0



Alpha=0.3



Math Details

- 3 BRDFs for different hardware/performance
- BRDF1: PC/consoles, SM3.0+
  - Derived from Disney; Torrance-Sparrow μfacet model
  - Blinn-Phong NDF, Smith V, Schlick F
- BRDF2: Mobile ES3.0+
  - Modified “Minimalist Cook Torrance”
  - Approximate Kelemen & Szirmay-Kalos V, approx F
- BRDF3: Old hardware (DX9 SM2.0, DX11 9.x, ES2.0)
  - Blinn-Phong in RDF form, implicit V, no F
  - Math done via lookup texture

How much perf does it cost?

- Small scene to test on mobile & PC
- No shadows, no post-fx; just raw cost of shader itself
- Render into 1920x1080 everywhere

How much perf does it cost?

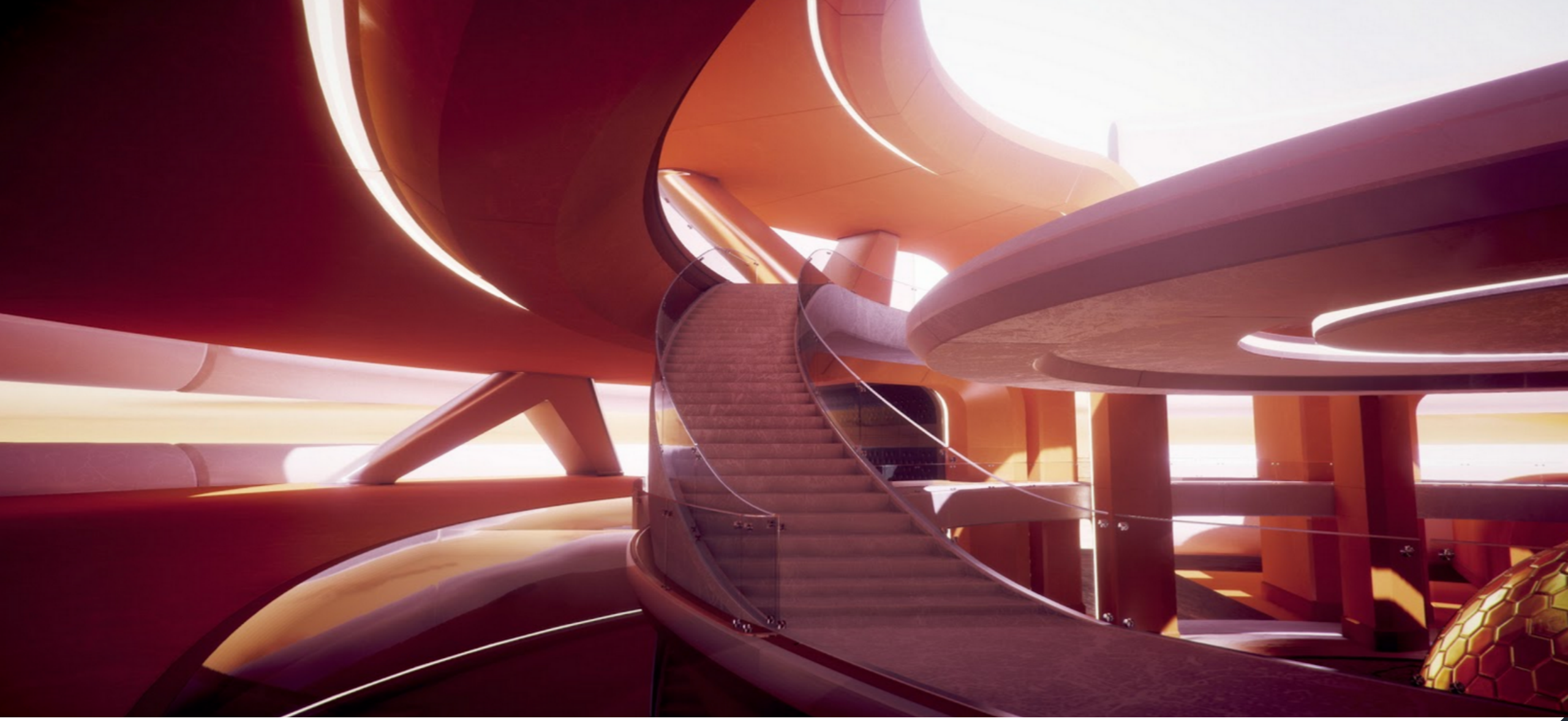
unity

How much perf does it cost?

- Added ms/frame on GPU compared to Unity 4 BumpSpecular:
  - Fast PC (GeForce GTX 680): +**0.1**ms
  - Medium PC (Radeon HD 7770): +**0.4**ms
  - Laptop (GeForce GT 750M): +**1.4**ms
  - Laptop (Intel Iris Pro): +**3.4**ms
  - iPhone 6 (A8): +**8.3**ms
  - iPadAir (A7): +**12.3**ms
- Not a fair comparison! Now it samples more textures (occlusion, reflection etc.)
  - Also you probably don’t want to render 1080p with PBS on older mobiles

Surface Shaders

- Can use Standard BRDF in surface shaders too
  - #pragma surface surf Standard
  - void surf (Input IN, inout SurfaceOutputStandard o) { … }
  - Similar for specular workflow
- Your own PBS lighting functions
  - “GI” part (lightmaps, ambient, indirect, reflections) is part of lighting function
  - Was hardcoded & non PBS in Unity 4



Global Illumination & Lightmapping

unity

Global Illumination in Unity 5

- Precomputed realtime GI
  - Lightmaps + Light Probes
- Baked Static GI
  - Lightmaps + Light Probes
- Reflection Probes
  - Baked & Realtime

Precomputed Realtime GI

- Lightmaps and probes updated realtime
  - Dynamically change light sources, emissive materials & environment lighting
- Static objects lit by realtime lightmaps
- Dynamic objects lit by realtime light probes
  - 2nd order spherical harmonics

Precomputed Realtime GI

- Objects that affect lighting are static
- Moving objects pick up GI, but do not affect it
- Diffuse light transport only
  - Final reflective bounce: directionality and/or cubemaps
- Low frequency / low resolution

Baked Static GI

- Ye olde lightmaps + light probes (3rd order SH)
- Baked direct + indirect + AO

Directional Lightmaps

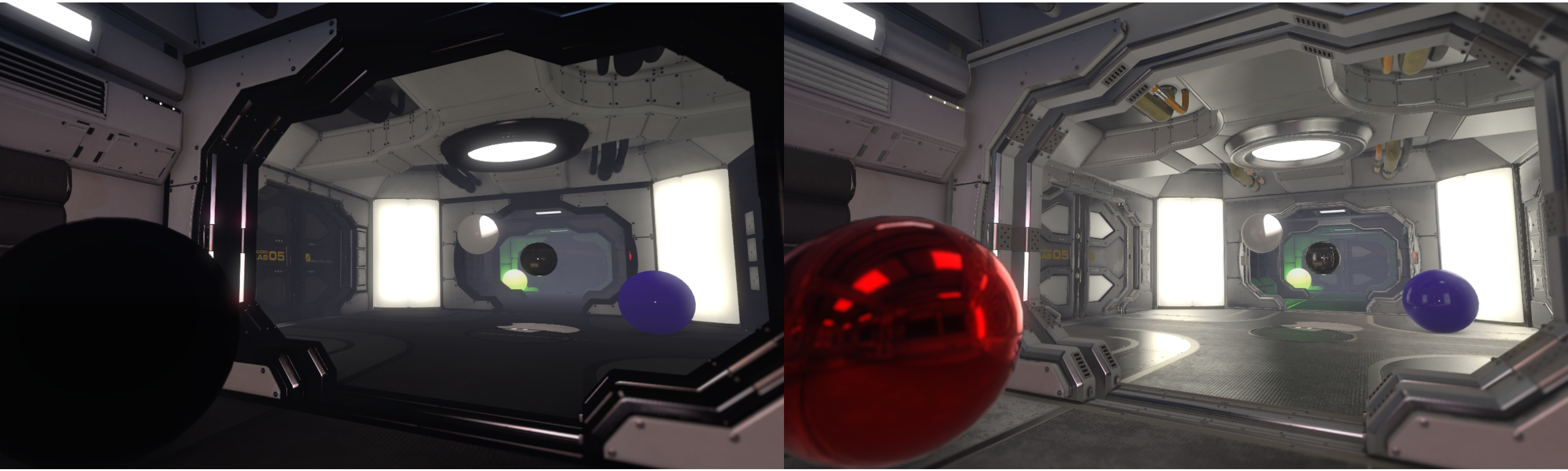
Simple

Directional

Directional Specular

Lightmapping Workflow

- Hash & cache
  - All inputs (meshes, transforms, lights, materials) hashed
  - Results of jobs cached
- Actual work done in async & parallel jobs
- Continuous baking mode
  - Monitors changes & automatically does stuff
- On-demand baking
  - Press a button
  - Lightmap snapshot saved & can be versioned



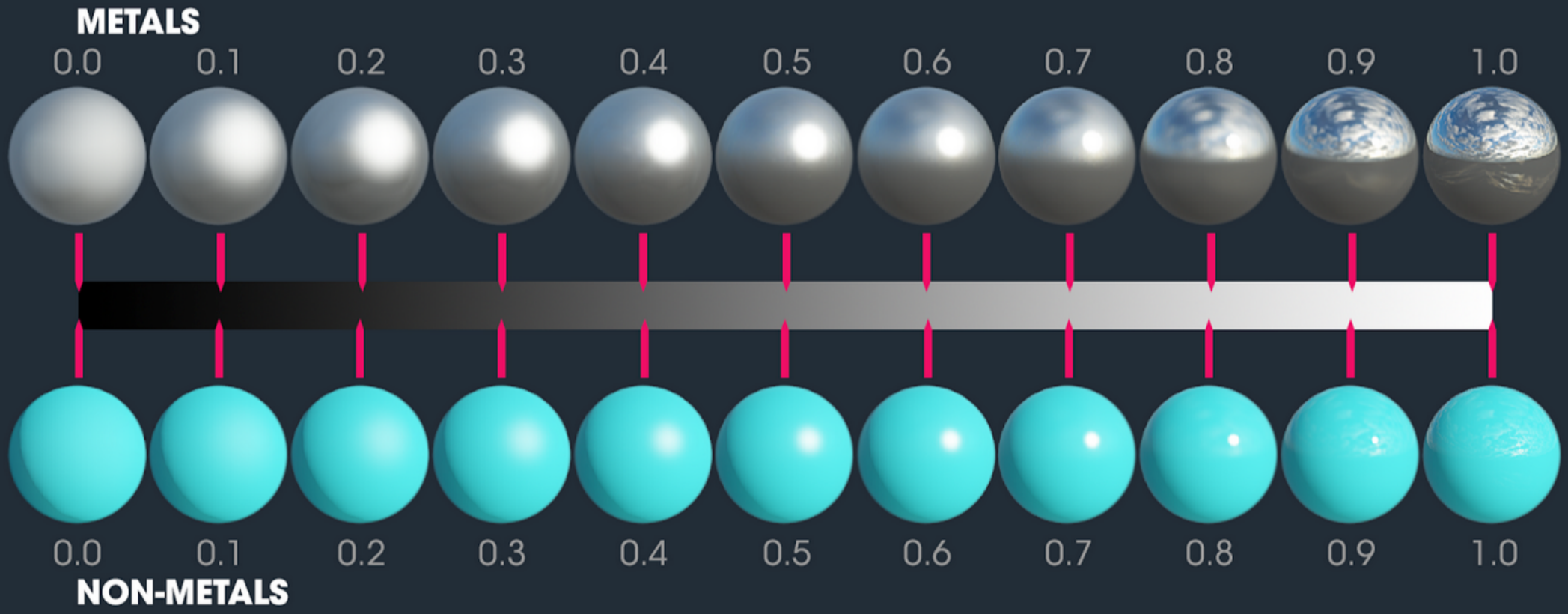
Reflection Probes

Reflection Probes

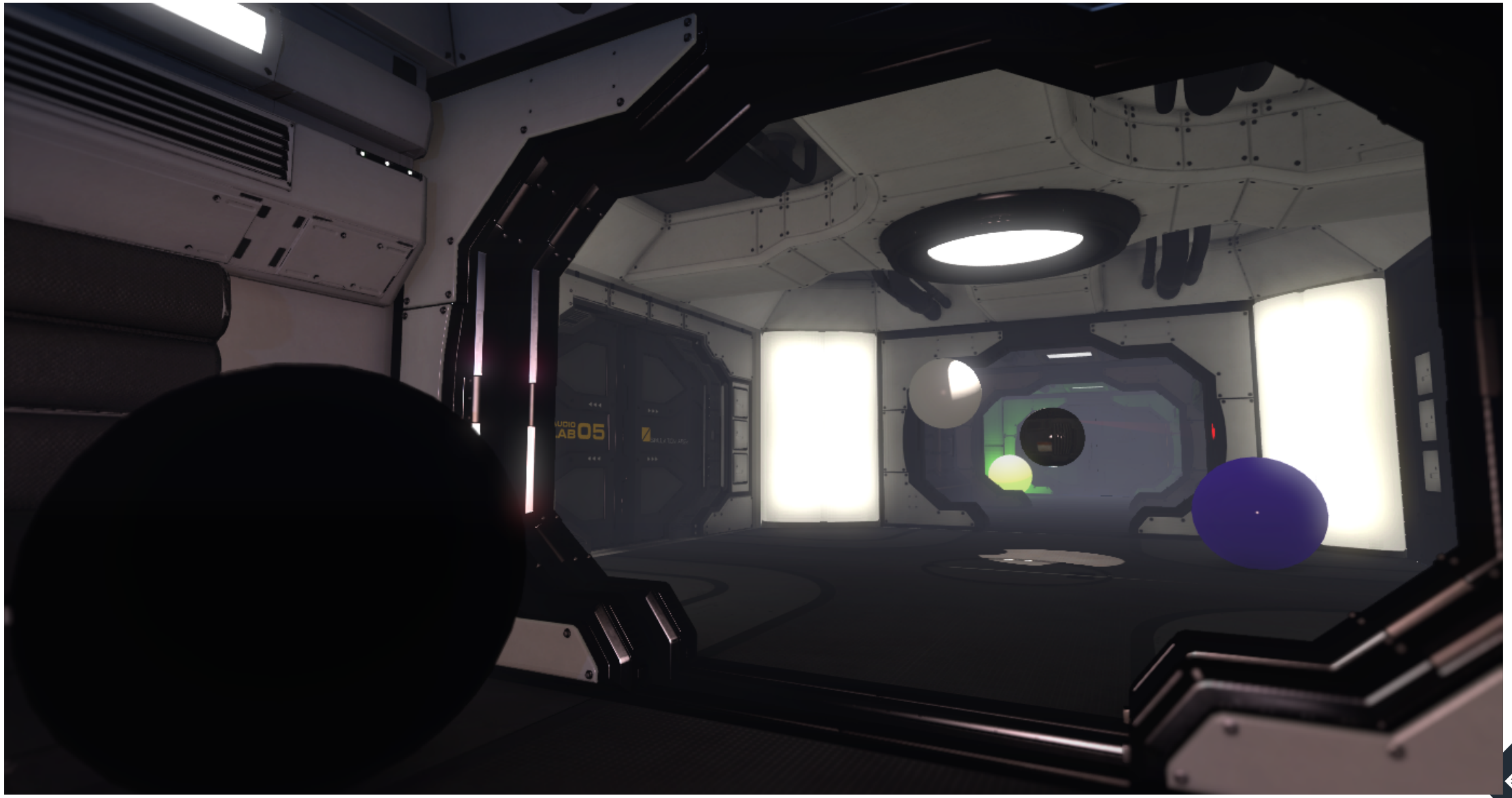
- Boxes placed in scene
- Each box captures a reflection cubemap
- Shader samples closest probe(s) for reflection
  - Mip levels for varying smoothness (convolved with Phong lobe)
  - Box projection
  - Blending between probes

SMOOTHNESS A

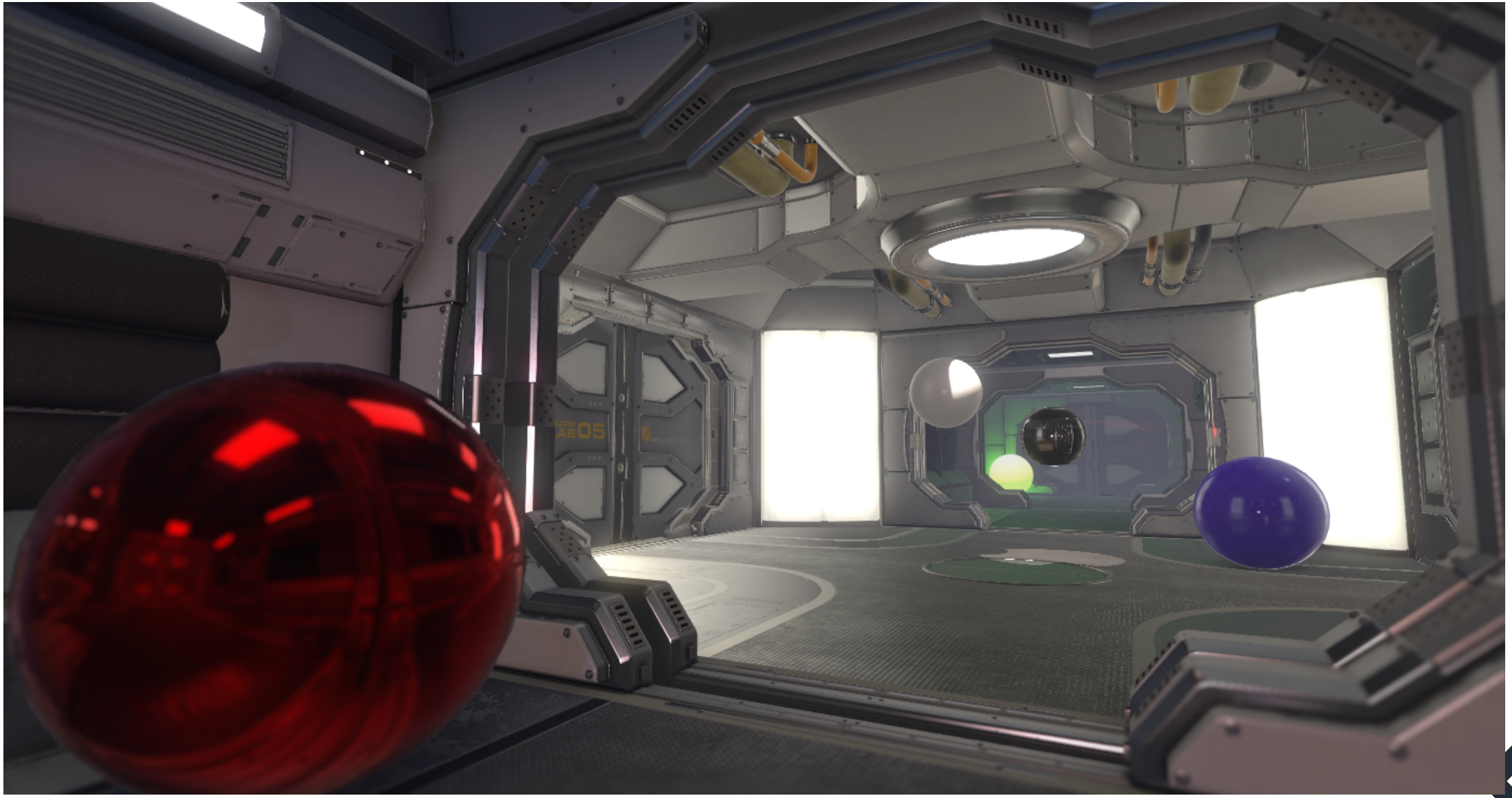
SMOOTHNESS DEFINES THE PERCEIVED **GLOSSINESS** OR **ROUGHNESS** OF A SURFACE
FOR TEXTURES, THIS IS STORED AS THE ALPHA CHANNEL OF THE **SPECULAR MAP**



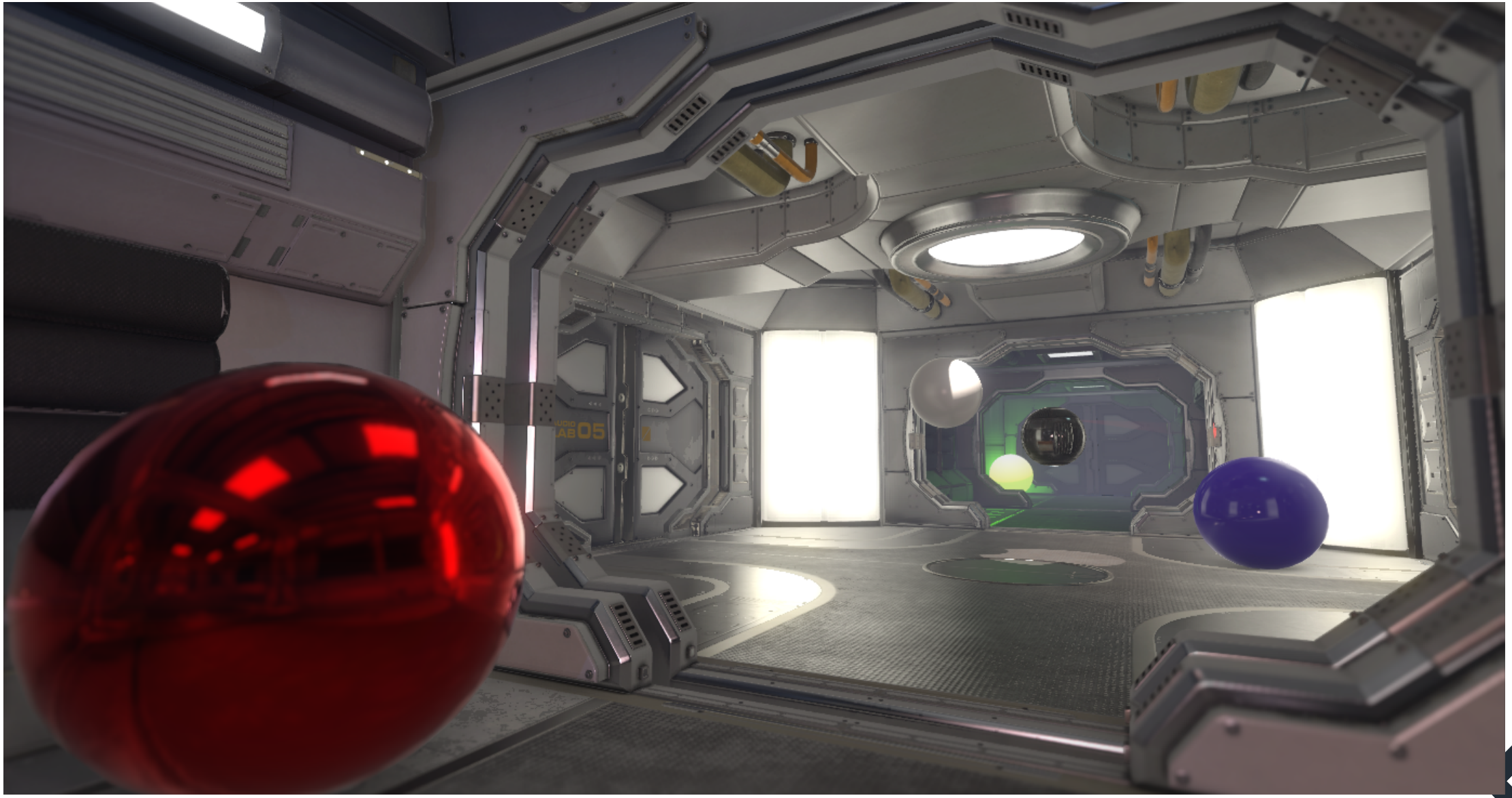
No Reflection Probes



Reflection Probes (no box projection)



Reflection Probes with box projection



Baked & Realtime Probes

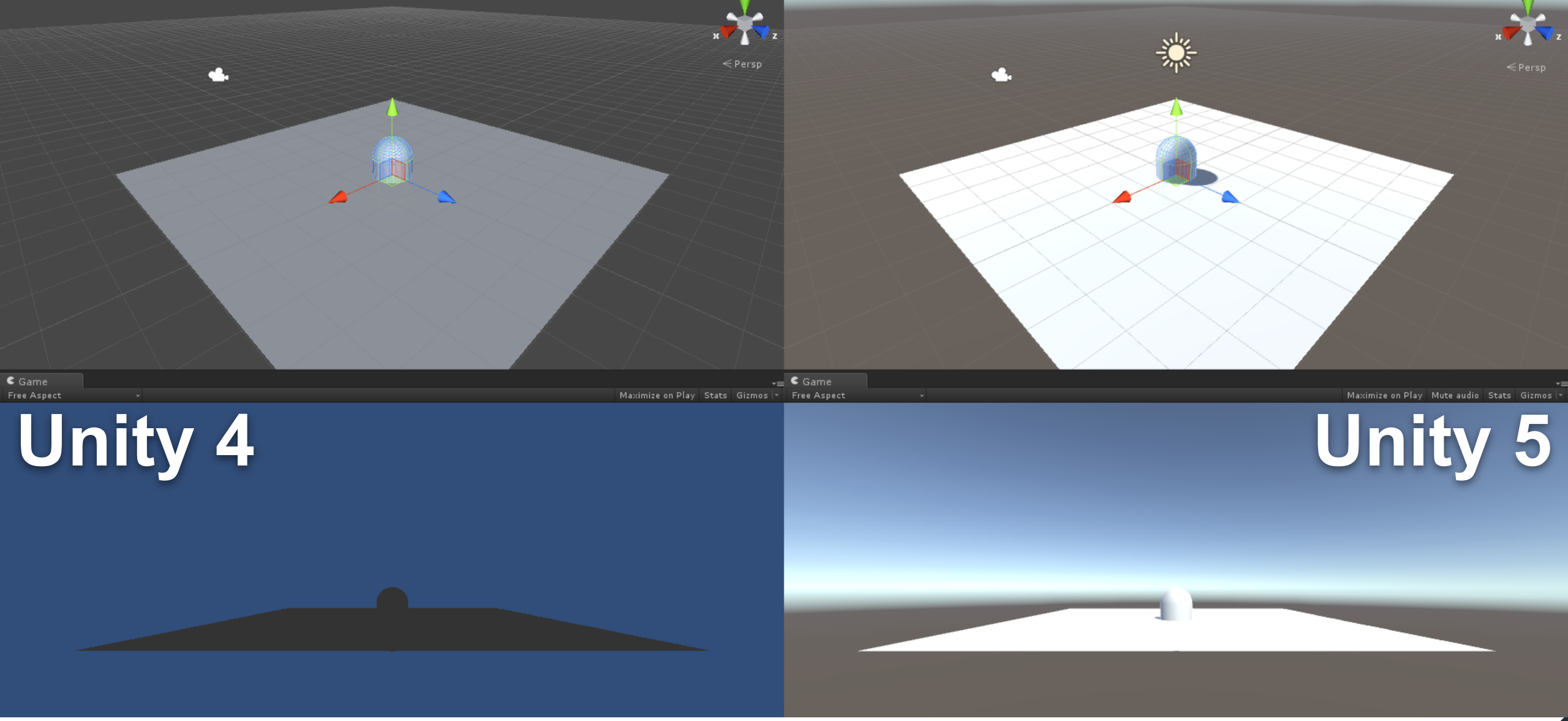
- Baked
  - Static
  - High quality
- Realtime
  - Useful for dynamic scenes / hero chars
  - Render scene into cubemap at runtime
    - Can be spread over frames
  - Lower quality mip convolution (on the GPU)
    - Can be spread over frames

Future: SSRR

- Screen Space Raytraced Reflections (not in 5.0)
- Current plan:
  - Deferred Shading only (need G-buffer info)
  - Split reflection probes buffer from emission/lightmaps buffer
  - Render probes similar to deferred lights
  - SSRR, fallback to probes where no information

No screen-space reflections

With screen-space reflections

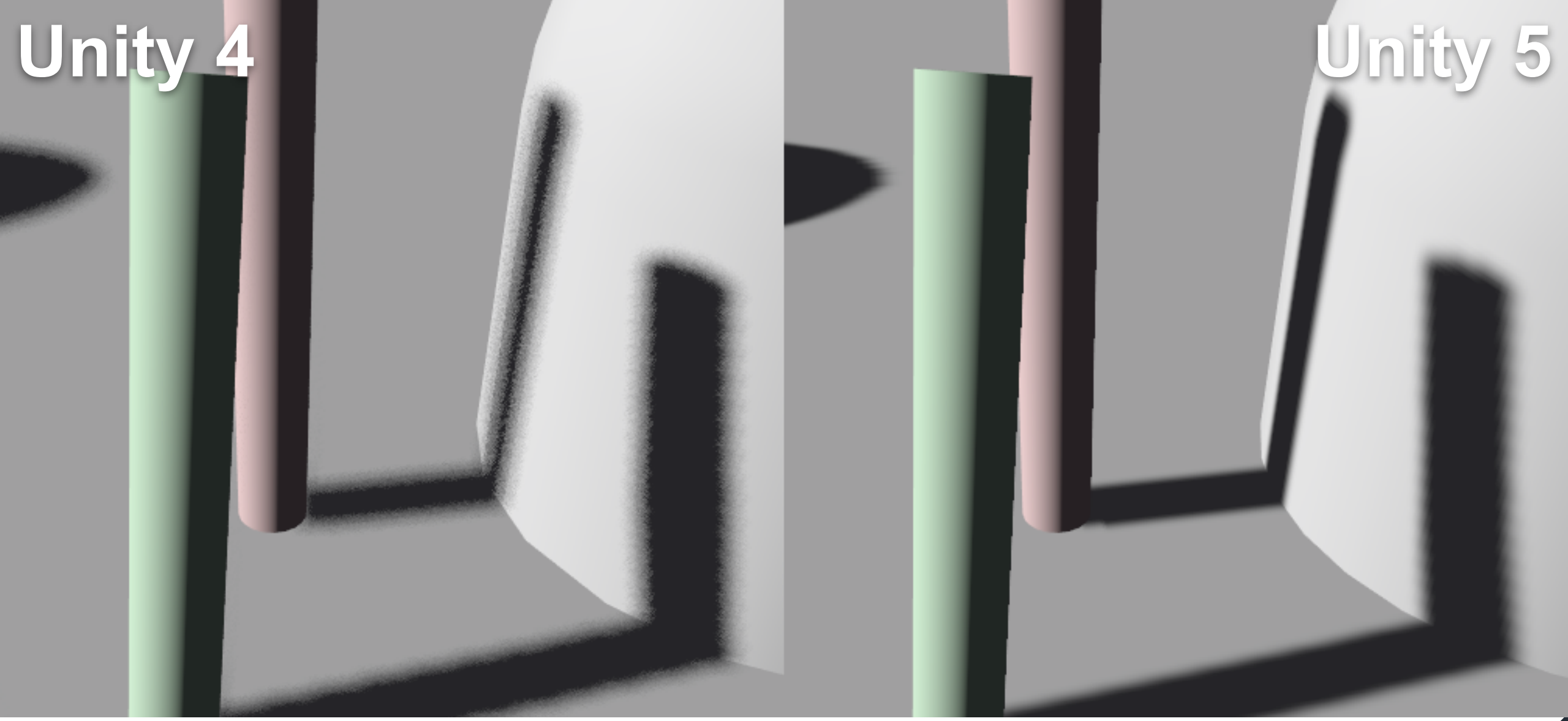


Workflow Improvements

unity

Little Things

- Procedural skybox shader
  - Skyboxes can be HDR too
  - HDR (.exr/.hdr) texture importing too
- New scene in 3D game has skybox & light
  - Ambient & reflection matching skybox
  - Scene view can be HDR too
- Cubemap workflow
  - Automatic detection of cross/latlong/strip/spheremap layouts
  - Texture compression



Shadows

unity

Directional Light Shadows

- PCF 5x5 instead of screenspace blur
  - Looks better & runs faster
  - Normal offset shadows
- Cascade split ratios & visualization
- No more shadow collector pass
  - Shadow mask computed from depth texture

Shadow cascades visualization

Other Shadow Stuffs

- Shadow casting modes
  - Off, On, Two Sided, Shadows Only
- Point/spot shadows no longer rendered 2x
  - If you have both deferred & forward objects
- Better bias for spot lights
- Better precision for point lights (RFloat format)
- Better point/spot filtering not in 5.0 yet, but high on the list

Deferred Shading

unity

Deferred Shading

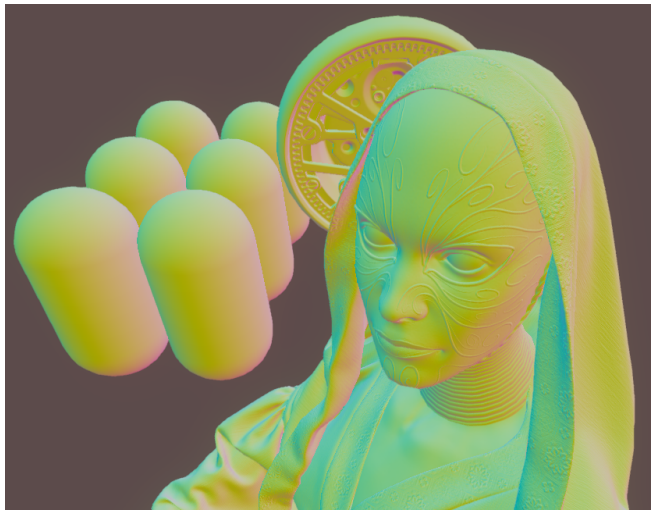
- Unity 4 Deferred Lighting can’t really do PBS
  - Not enough space in G-buffer to store information
  - Two geometry passes aren’t nice either
- Unity 5 adds deferred shading
  - Multiple Render Targets to store G-buffer
  - One geometry pass

Deferred G-buffer

- Four render targets, 160bpp (LDR) or 192bpp (HDR)
  - RT0: diffuse color (rgb), occlusion (a)
  - RT1: specular color (rgb), smoothness (a)
  - RT2: world normal (rgb, 10 bit/channel)
  - RT3: emission/light buffer; FP16 when HDR
  - Z-buffer: depth & stencil

Diffuse

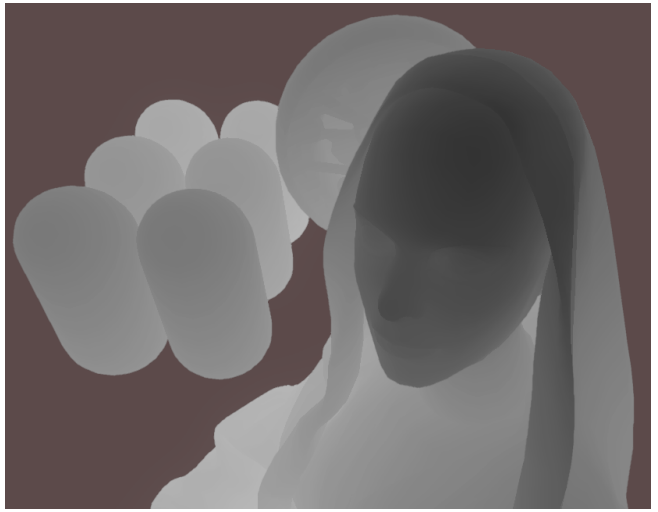
Normals



Specular

Smoothness

Depth



Final result

More Consistency

- Linear & Gamma color spaces more similar now
  - Lightmaps almost the same
  - More similar light intensities & specular highlights
- Removed hidden “* 2” on light intensity in shaders
  - Custom shaders might need an update in Unity 5
  - Also code or animations that work with light color/intensity

Meshes

- Can haz 4 UV coordinates
  - Up from 2 UVs in Unity 4
  - Vertex colors can be floats (Mesh.colors vs colors32)
- No CPU/memory cost for non-uniform scale
  - Shaders do transform / normalization
- Per-Renderer instance mesh data
  - e.g. painting vertex colors in the world
  - MeshRenderer.additionalVertexStreams

Shaders: Stripping

- Build-time stripping of unused shader variants
- #pragma shader_feature
- Unused fog / lightmap modes
- Standard shader can have 35k variants in total
  - 375MB if compiled for DX11 only (26MB zipped)
- Most content ends up using several hundred
  - Several megabytes

Shaders: Load Performance

- Shader loading performance
  - Generally loads less variants than Unity 4
- ShaderVariantCollection to control preloading
  - List of shaders + their variants
  - Can be recorded from editor
  - Replaces the very blunt WarmupAllShaders

Shaders: Fog

- Fog done differently
- Works on all platforms now! (WP8 & consoles)
- Shader macros to do fog
  - Instead of Unity 4 runtime shader patching

Other Misc Stuff

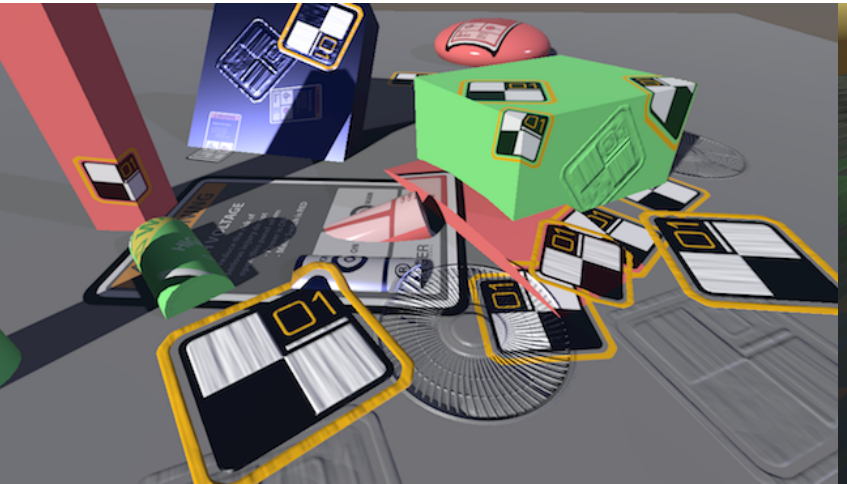
- More Performance
  - Light culling in forward rendering
  - Shadow caster culling
  - Less SetPass calls, especially with light probes
- More shader keywords
  - 128, up from 64

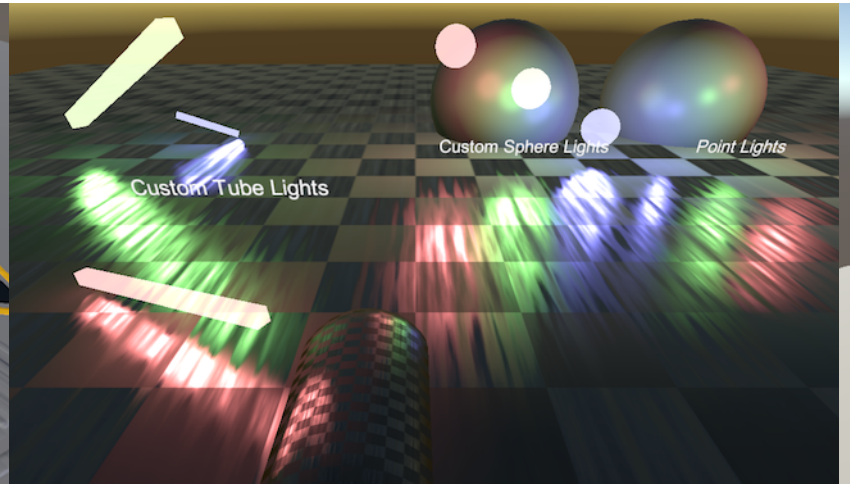
Extensibility: Command Buffers

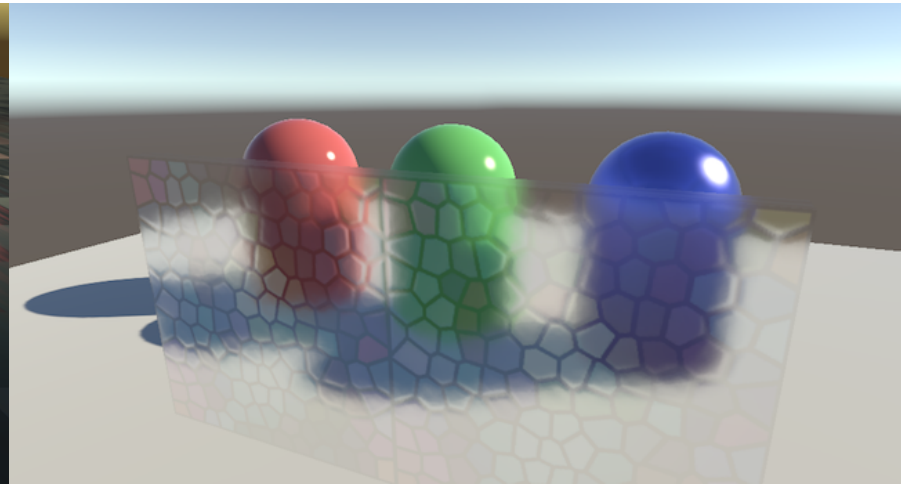
- Create “lists of things to do”
  - Draw mesh, set render target, blit with shader, …
- Tell cameras to execute them at various points
  - After deferred G-buffer
  - After deferred light pass
  - Before all transparencies
  - …

Things to do with command buffers

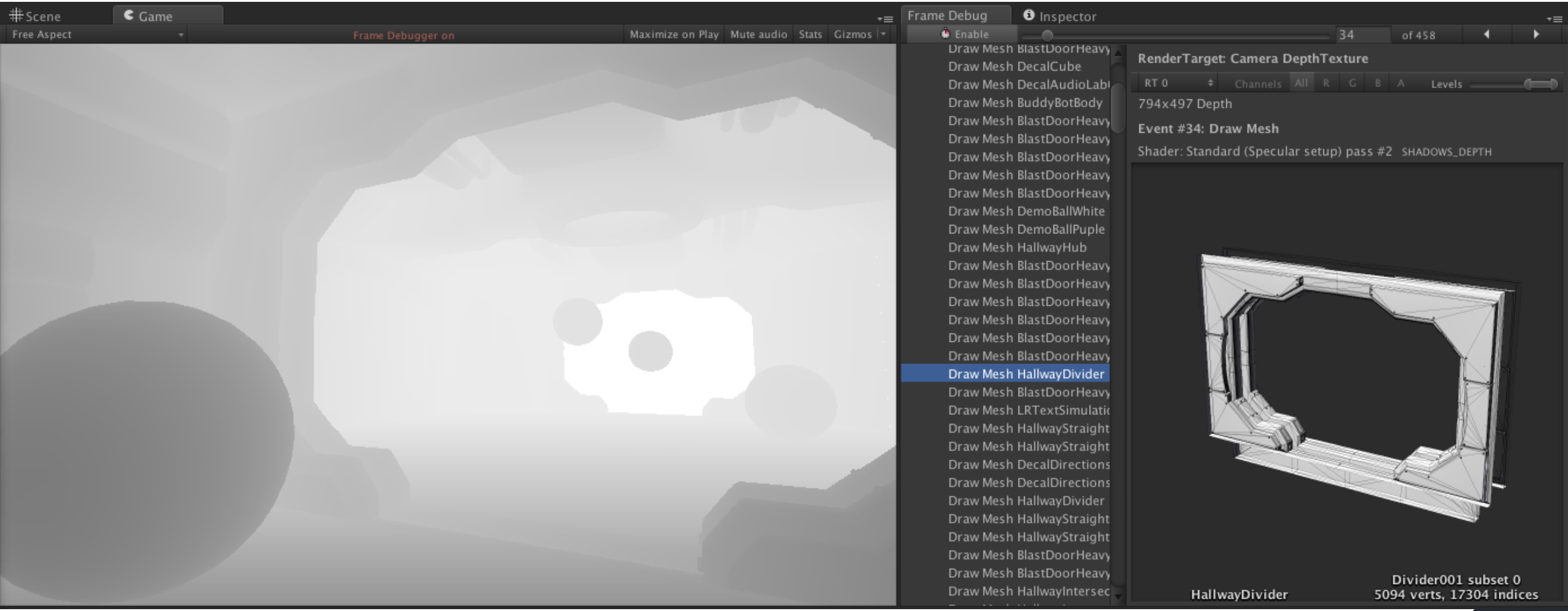
- Deferred decals
- Custom lights in deferred shading
- GrabPass on steroids
- …



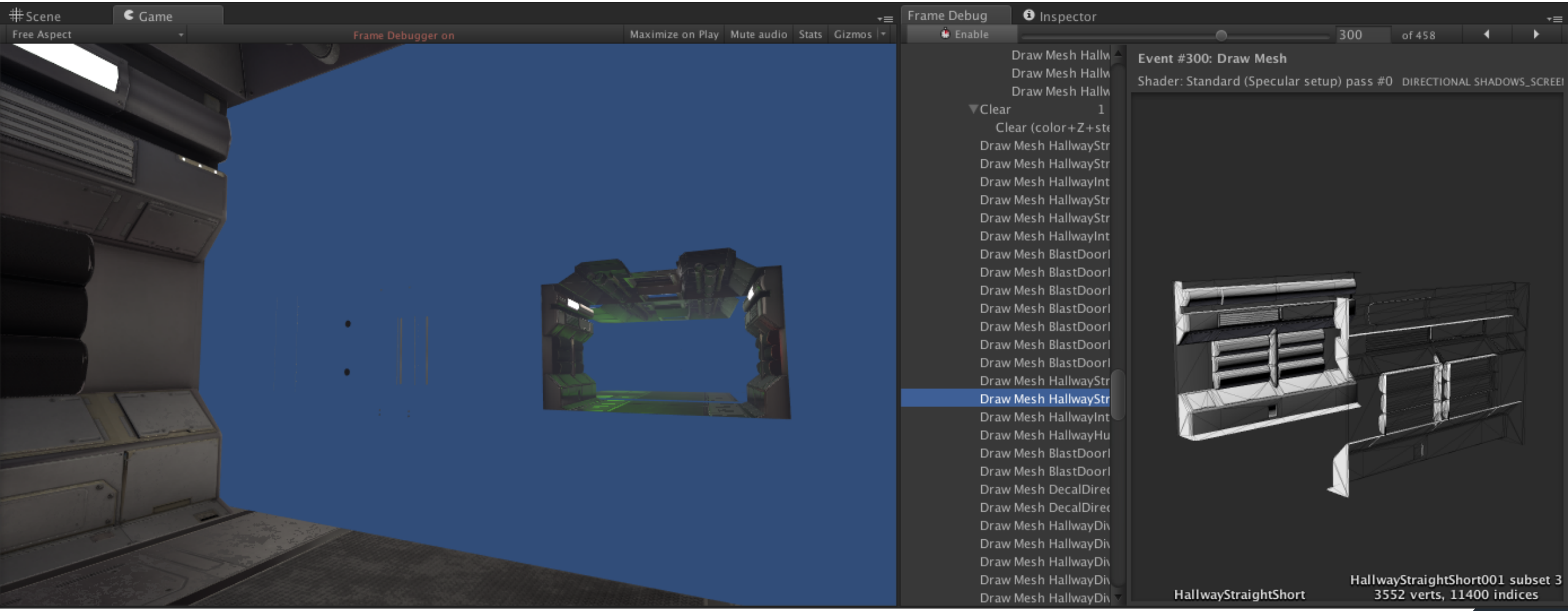




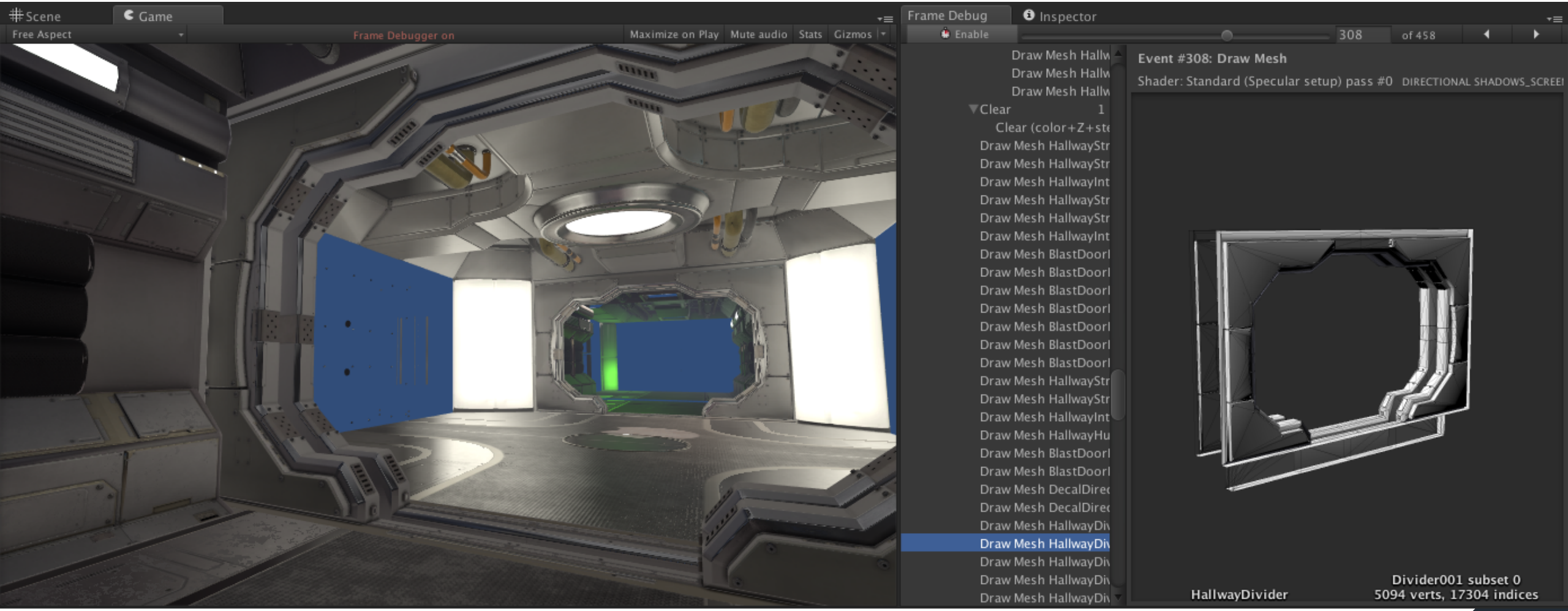
Frame Debugger



Frame Debugger

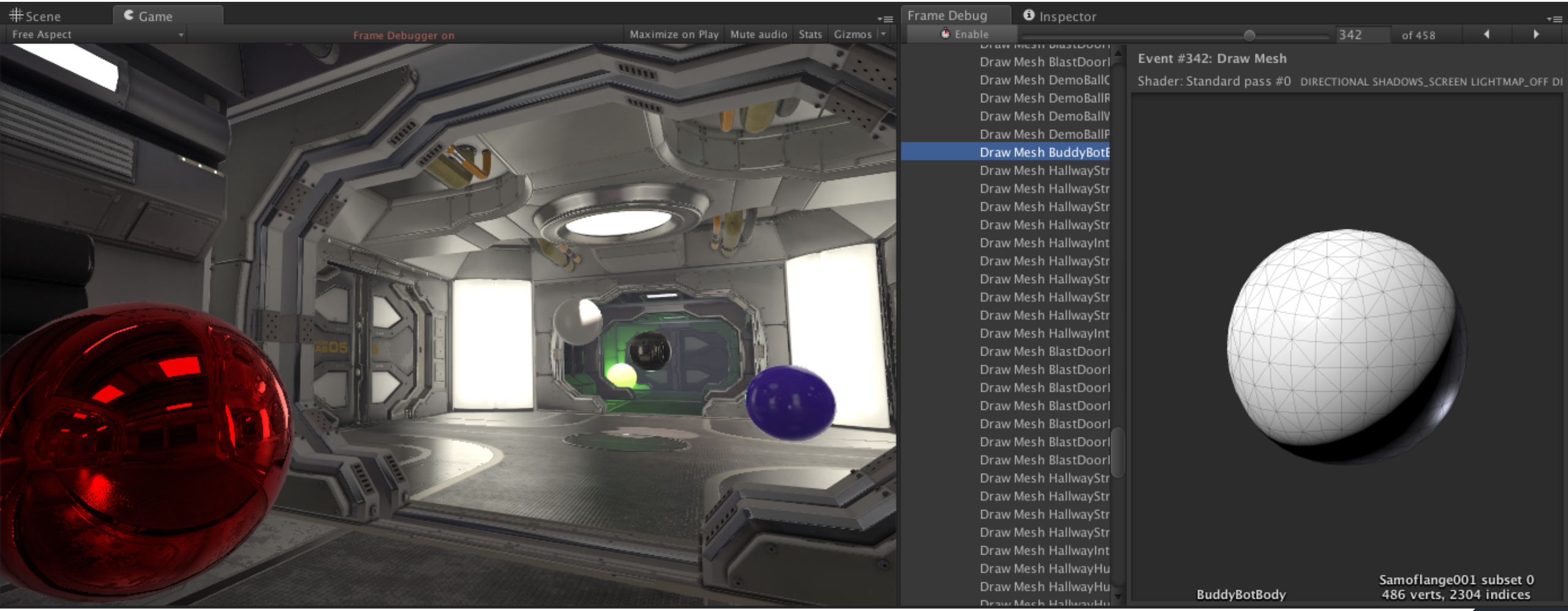


Frame Debugger



unity

Frame Debugger




Questions?